I

SOFTWARE DE INTERFACE PARA BANCO DE DADOS

Carlos Alberto de Castro e Abreu

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA O OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.)

Aprovada por:

José Lucas Mourão Rangel Netto
(Presidente)

João Lizardo R.H. de Araujo

Paulo Augusto Veloso

William Carlyle Koelsch

RIO DE JANEIRO, RJ - BRASIL

JANEIRO DE 1978

RESUMO

Trata do relacionamento usuário - banco de dados.

O STELA, sistema de interface entre o usuário e um sistema de gerência de banco de dados, no caso o TOTAL, vem sendo projetado e construído há cerca de cinco anos pela PUC R.J. tendo sido parcialmente financiado pelo BNDE.

Os seus componentes são: a seleção, a impressão, a ordenação, o cálculo e a tabulação.

O projeto, as especificações técnicas e a programação do módulo de tabulação, bem como a interação com os demais módulos, constituem o objetivo precípuo desta tese.

São ilustradas as necessidades do uso da tabulação.

A descrição analítica do projeto das várias etapas, necessárias ao seu funcionamento, são também discutidas.

A lógica interna de programação é explicada através de fluxos de lógica.

Os programas em linguagem Assembler usados na implementação estão inseridos nos apêndices.

SUMMARY

This work presents a module of the STELA system which acts as the interface between the user and a database (TOTAL).

The STELA system was developed by Pontifical Catholic University of Rio de Janeiro (PUCRJ), with financial support from the National Bank for Economic Development (BNDE).

The STELA system is made up of five modules, namely selection, printing, ordering, calculation and cross-reference. This works concerns itself in particular with the last one, and its interaction with the other modules.

We present here the description and the internal logic of the several parts of which the module is composed. The actual assembly language listings of the programs appear in the appendix.

AGRADECIMENTOS

Ao Prof. William Carlyle Koelsch, meu orientador na PUC e responsável pelo projeto STELA.

AGRADECIMENTOS

Ao Dr. Claudio Portela Peixoto, chefe do Departamento de Sistemas do BNDE, cujo apoio para estudos e pesquisas jamais nos faltou.

Ao Prof. José Lucas Mourão Rangel, meu orientador na COPPE.

A toda a equipe de desenvolvimento do STELA formada pelas seguintes pessoas: Marcos A.N. Ferreira, Rosaura Maria C.L. Eichenberg, Leonardo L.P. Leite, Renato J.S. Bahia, Flavio Mariano A.P. de Carvalho e Marco Antonio Romero , pelos esclarecimentos prestados durante os trabalhos de pesquisa.

A Carlos Marques Olivieri companheiro de projeto do BNDE pela ajuda na discussão da tese e depuração dos programas.

A Denise, minha esposa, pela revisão ortográfica dos textos.

Capítulo I - INTRODUÇÃO

O âmbito desta pesquisa trata do relacionamento usuário-banco de dados. Várias transformações são necessárias para que um pedido do usuário crie um conjunto de especificações técnicas de máquina que provocam a busca nos bancos de dados e a posterior exibição do resultado.

Para possibilitar uma plena utilização do diálogo, é necessário que o processo seja conversacional, interativo e que possua tempo de resposta conveniente.

Este projeto de pesquisa vem sendo elaborado há cerca de cinco anos pela Pontifícia Universidade Católica - PUC R.J. e financiado em grande parte pelo Banco Nacional de Desenvolvimento Econômico - BNDE. Os recursos computacionais necessários à implementação foram os da própria PUC, tendo sido usado um IBM 370 Modelo 165 sob o sistema operacional OS -MVT e utilizando o método de compartilhamento de tempo TSO da IBM. As características de arquitetura deste computador: palavras de 32 bits, 16 registradores gerais e representação hexadecimal do código binário de máquina, influiram decisivamente na elaboração dos programas.

Foi necessário também a escolha de um sistema de gerência de banco de dados que pudesse garantir a manutenção e integridade das informações. Foi escolhido então, o TOTAL, construído pela CINCON SYSTEMS em 1968, por sua ampla utilização no país e exterior (cerca de 1400 instalações no mundo até junho de 1976) (1).

Alguns princípios gerais dos sistemas de gerência de banco de dados, bem como as características específicas deste software são assuntos do CAP III.

Interagindo entre o usuário e o banco de dados existe uma interface que é o STELA explicado em linhas gerais no capítulo II. Ela possibilita que um pedido do usuário seja transformado, através da geração automática de tabelas e blocos de controle, em programa executável de máquina, orientado para busca de arquivo TOTAL.

O sistema operacional próprio do STELA controla as

interrupções entre as tarefas do sistema; executa o controle de linha dos terminais e controla a paginação por software, usando blocos de controle específicos.

Para uma maior facilidade de utilização do sistema, todos os módulos paginàveis do STELA são reentrantes, podendo a mesma cópia ser usada por vários usuários. As áreas de dados geradas automaticamente são colocadas em páginas independentes de 4 k bytes que permitem a segmentação dos módulos, visando um uso econômico de memória principal.

Podemos dizer, resumindo, que o STELA consiste de um sistema de recuperação de informações usando técnicas de teleadministração de bancos de dados conforme ilustra a figura 1 na página 31.

Para maior facilidade de uso, o STELA vem sendo projetado e construído de uma forma modular. Existe um conjunto de funções de tratamento e de recuperação de dados, que trabalham independentemente, porém supervisionadas pelo núcleo do sistema. Estas funções são: a seleção, a impressão, a ordenação, o cálculo e a tabulação, sendo chamadas por verbos específicos.

O projeto, as especificações técnicas e a programação do módulo de tabulação, bem como a interação com os demais módulos do STELA, constituem o objetivo precípuo desta tese.

A necessidade do uso da tabulação, bem como explicações a nível de usuário do seu funcionamento, são tratadas no capítulo IV.

O capítulo V descreve de uma forma detalhada a abrangência da tabulação, bem como todas as etapas necessárias para a transformação dos pedidos do usuário em programa executável.

Mais adiante no capítulo VI são analisados através de fluxos, a lógica interna das várias etapas. Posteriormente encontram-se a conclusão final, os apêndices e as referências bibliográficas.

Implementamos em linguagem Assembler IBM 370, e usa

mos uma versão fornecida à PUC pela Universidade de Waterloo-Canadá, através de convênio, para manter a compatibilidade com o restante do projeto (3) (12). Visamos com isto, obter também o menor código objeto possível, fazendo pleno uso dos recursos internos de máquina.

Capítulo II = VISÃO GERAL DO STELA

Os módulos componentes trabalham sob a supervisão do Executivo que concentra as funções de suporte (10). Ele consiste de um conjunto de rotinas de uso geral e de construção modular, porém de execução interdependente. Suas funções são: a gerência de memória, a gerência de processos e compartilhamento de tempo, acesso ao banco de dados e controle da rede de terminais.

O projeto objetivou uma baixa utilização da memória principal para permitir uma futura utilização em sistemas de porte médio. Além disso, para possibilitar o máximo aproveitamento da memória disponível, a estruturação dos recursos de memória foi feita de forma similar a sistemas de memória virtual.

As características básicas de gerência de memória são:

- O núcleo é integralmente residente.
- Os módulos de aplicação são codificados de forma modular e como se houvesse uma área virtualmente ilimitada para contê-los.
- Os dados são colocados em áreas específicas, fisicamente separados dos programas.
- Um conjunto de áreas de dados é atribuído a cada usuário, que é representado por um terminal interagindo com o sistema.
- O STELA é portanto, totalmente reentrante.

Durante a execução de uma página de aplicação, o máximo de gasto de memória é 46 k bytes assim discriminados: 4k bytes da própria página, o conjunto de páginas da área de dados cujo número máximo são 4 totalizando 16k, além do núcleo cujo tamanho é de 26k bytes.

Ao gerente de processos cabe salvar as informações básicas por ocasião de interrupções. Também é responsável pela transferência de controle entre os vários módulos de um mesmo processo.

O sistema usa memória de disco para manter o arquivo das páginas (STELPAGE) e para áreas auxiliares de trabalho

das aplicações (STELWORK). O acesso a disco é realizado por um método de acesso do próprio Executivo que garante uma única operação de entrada e saída para efetuar todas as leituras e gravações de uma mesma trilha, numa única rotação, através da criação dinâmica de programas de canal. Os pedidos de acesso são grupados na ordem de endereço físico de disco, procurando minimizar o movimento de braço e o próprio uso do espaço em disco é feito de forma a aumentar a probabilidade de ocorrer acesso a mais de um bloco na mesma trilha. Visando otimização, os acessos a arquivos em disco em unidades diferentes, podem ser realizados em paralelo através da execução simultânea de vários programas de canal.

Outro objetivo no projeto foi tornar as aplicações, independentes da máquina e configuração. Isto foi conseguido, por intermédio de uma interface padrão para realizar as operações de entrada/saída externas. Esta interface se comunica com a rotina de acesso que é específica para cada tipo de configuração.

Após a instalação do sistema, os vários ambientes e configurações com os quais o STELA vai operar, são definidos pela geração de módulos de carga chamados genericamente de CONFIGURAÇÃO e que, além de descrever a rede de terminais, contém os blocos de controle e áreas de trabalho cujo tamanho e quantidade sejam dependentes da configuração propriamente dita. A especificação de um parâmetro em tempo de execução do STELA identifica qual é a configuração a ser carregada na memória. Os métodos de acesso são carregados posteriormente em função dessa configuração.

A arrumação dos módulos em páginas é feita por um sistema de referência cruzada, para facilitar o endereçamento às páginas durante a execução de uma aplicação.

Para possibilitar uma visão geral do sistema, é mostrado um fluxo do mesmo na fig. 2 da página 32.

Capítulo III = PRINCÍPIOS BÁSICOS DE UM SISTEMA DE GERÊNCIA DE BANCO DE DADOS

CARACTERÍSTICAS GERAIS DO SISTEMA TOTAL

Vamos examinar agora como um programa de aplicação, qualquer, interage com um sistema genérico de gerência de banco de dados (4). Vamos analisar as várias ocorrências necessárias para que o programa leia um registro de banco de dados , por meio do sistema gerenciador.

1- O programa de aplicação chama o sistema de gerência de banco de dados para ler um registro. Este programa indica o nome dado pelo programador para cada campo e informa o valor do campo chave do registro em questão.

2- O sistema de gerência de banco de dados obtêm a descrição dos dados do programa, que é usada pelo programa de aplicação e pesquisa a descrição dos mesmos.

3- O sistema de gerência de banco de dados obtêm a descrição de dados lógicos globais e determina que campos lógicos são necessários.

4- O sistema de gerência examina a descrição do banco de dados físico e determina que registro ou registros físicos devem ser lidos.

5- O sistema de gerência dá um comando para o sistema operacional do computador, instruindo-o para ler o registro requisitado.

6- O sistema operacional interage com o dispositivo físico aonde o dado se encontra.

7- Os dados requeridos são transferidos da memória intermediária para os buffers do sistema.

8- Comparando a descrição de dados do programa com a descrição dos dados lógicos globais, o sistema de gerência de banco de dados monta o registro lógico necessário ao programa de aplicação. Nor-

malmente as transformações de dados são tratadas nesta fase, pelo sistema gerenciador.

9- O sistema de gerência transfere o dado dos buffers do sistema para a área de trabalho do programa de aplicação.

10- O sistema gerenciador provê informação de "status" para o programa de aplicação, incluindo qualquer indicação de erros.

11- O programa de aplicação pode então operar com os dados na sua área de trabalho.

Para ilustrar melhor estas 11 fases, elas são sintetizadas no fluxo constante da figura 3 na página 33.

Características Gerais do Sistema de Gerência de Banco de Dados TOTAL

Possui a característica de independência de dados ou seja desvinculação completa entre dados e programas que os manipulam. Além disso, é permitido o acesso independente a um conjunto qualquer de dados do registro, de forma que apenas os dados necessários ao programa de aplicação são recuperados(6).

A comunicação entre o programa e o TOTAL é realizada através de CALL, em que um conjunto de parâmetros são passados.

As estruturas de dados em árvore ou hierárquica tipo IMS são as mais populares (7), porque foram as primeiras a serem usadas. Não obstante, o TOTAL apresenta uma estrutura em rede de relativamente fácil implementação, possuindo dois tipos básicos de arquivos.

O arquivo mestre possibilita o armazenamento do registro através da randomização da sua chave através de um algoritmo de "hashing" que gera um endereço relativo do registro no arquivo. A ocorrência de sinônimos pode ser melhorada, aumentando-se neste caso, o espaço disponível do arquivo.

O outro tipo de arquivo chamado variável, permite o registro de múltiplas ocorrências de informações, pertencentes ao arquivo mestre.

O conjunto de registros do arquivo variável ligado, a um arquivo mestre é chamado de cadeia e é estruturado através de uma lista duplamente encadeada.(8)

A ligação entre o mestre e a cadeia do variável é realizada através de 2 ponteiros, o primeiro apontando para o primeiro registro da cadeia e o segundo para o último registro.

A chave do registro mestre é repetida no registro do variável e não existe ponteiro do variável para o mestre.

Para melhor estruturação da rede é permitido que um variável tenha ligação com mais de um arquivo mestre.

Os registros variáveis são armazenados serialmente

e cada registro poderá ter múltiplas chave de acesso, uma para cada mestre ao qual está associado. A ligação entre os arquivos mestres e variáveis é feita através da chave de acesso presente nos dois arquivos e de um campo de ligação chamado "linkpath".

No arquivo mestre, este campo se subdivide em duas informações. Os primeiros quatro bytes indicam o primeiro registro na cadeia de registros variáveis, ligados a esse mestre através da chave. Os últimos quatro bytes indicam o último registro da cadeia.

Nos arquivos variáveis, o campo linkpath contém também duas informações: o indicador para o registro variável imediatamente anterior e o indicador para o registro variável, imediatamente posterior.

A definição dos arquivos, registros e campos é realizada pelo programa DBGEN que gera uma tabela contendo as descrições dos arquivos, a localização dos buffers, os nomes e tamanhos dos campos nos arquivos. Esta tabela está sempre disponível ao TOTAL, pois através dela é que são feitas os acessos ao banco de dados (11).

Consequentemente o TOTAL atinge os principais objetivos de um sistema de gerência de banco de dados:

1- O sistema deve permitir um método simples e padronizado de acesso a dados e atualização.

2- Segurança e integridade dos dados no banco devem ser absolutas.

3- A capacidade para expansão do sistema sem perda dos conteúdos presentes do banco de dados deve poder ser prevista no projeto do sistema.

4- O sistema deve utilizar eficientemente os recursos de hardware do computador. Deve requerer um uso pequeno de memória e usar pouco tempo de computação. Deve ter a capacidade de facilitar a eliminação de dados redundantes.

Como o sistema STELA não realiza nenhuma alteração no banco de dados, apenas extraindo as informações e apresen-

tando-as em vários formatos, as funções de criação, eliminação e atualização de registros não são usadas. O sistema utiliza somente funções de inicialização/término e funções de recuperação de dados (11).

As funções de inicialização e término utilizadas são as seguintes:

1- Funções de inicialização e término do sistema

1.1. TOTAL - função que deve ser realizada no pedido de chamada inicial, SIGN-ON, para indicar ao TOTAL o nome do banco de dados utilizado e o nome do usuário que pede entrada no sistema.

1.2. DEQUE - função que retira da tabela de usuários do TOTAL, o usuário em questão.

1.3. ENDTO - função que finaliza o funcionamento do TOTAL liberado e fechando automaticamente todos os arquivos.

2- Funções de inicialização dos arquivos

2.1. OPENM - função que realiza a abertura dos arquivos mestres.

2.2. OPENV - função que realiza a abertura dos arquivos variáveis.

2.3. RESTM - função que recoloca no início dos arquivos mestres os indicadores que determinam a leitura serial.

As funções CLOSM e CLOSV para fechamento individual de cada arquivo não são utilizadas. Os arquivos são fechados apenas quando se emite a função ENDTO para finalizar todo o sistema.

As funções de recuperação de dados são:

1- SEQRM - função que realiza a leitura serial dos registros do arquivo mestre. Os registros são extraídos conforme a ordem física de sua gravação, sendo desconsiderados os espaços em branco.

2- READM - função que realiza a leitura direta de um registro do arquivo mestre através de sua chave.

3- READV - função que realiza a leitura sequencial das cadeias de registros variáveis associadas aos registros mestres. Para realizar esta leitura, o usuário deve ter um conhecimento prévio da chave do registro mestre ao qual a cadeia de variáveis está associada e do nome do linkpath.

4- READD - função que realiza a leitura direta de um registro variável através de um indicador de localização interna que, como parâmetro das funções relacionadas a arquivos variáveis, é conhecido como REFER.

No STELA cada usuário tem reservada uma área chamada bloco de controle do usuário (UCB) onde são armazenadas todas as informações necessárias para que possa utilizar o sistema. É nesta área que se encontra a tabela de parâmetros que são passados para o sistema TOTAL. Trata-se de uma tabela de endereços que indica as localizações das várias informações, necessárias para a execução da função.

Capítulo IV - AS NECESSIDADES DO MÓDULO DE TABULAÇÃO

Existe uma procura significativa para tabelas de cruzamento. Uma tabela de cruzamento pode ser uma classificação de uma ou duas dimensões (13). Alguns exemplos de classificações de uma dimensão ou classificação simples são:

- Número de projetos por estado
- Número de projetos por ano da data do convênio
- Total de valor global dos projetos por setor
- Total das liberações previstas por mês no ano de 1970

Alguns exemplos de classificações de duas dimensões ou classificações duplas são:

- Número de projetos por estado e por setor
- Número de projetos por ano da data de convênio e por ano da data de término
- Total das liberações e amortizações previstas por mês e por fonte de recursos.

Existem duas maneiras diferentes de efetuar uma dessas classificações. A primeira é através do uso de verbos Selecione, Ordene, Calcule e Imprima. O usuário seleciona as fichas desejadas; ordena por campo (ou campos) pivotal; conta, totaliza ou faz outro cálculo aritmético em relação ao campo pivotal, e finalmente imprime os resultados.

A segunda maneira é através dos verbos Conte e Totalize. Esses verbos permitem uma seleção mas não constroem uma lista orientadora; fazem uma só pesquisa sequencial, construindo uma tabela na memória à medida que se encontra valores pivotais. Esses verbos funcionam unicamente para cruzamentos do tamanho limitado. Cruzamentos grandes exigirão o processamento mais complicado de seleção, ordenação, cálculo e impressão.

EXEMPLOS

Comando: CONTE PROJ POR ESTADO

Resposta: EFETUANDO UMA PESQUISA SEQUENCIAL NO ARQUIVO PROJ
ENTRE # PARA INTERROMPER E CANCELAR A PESQUISA
ENTRE ? PARA SABER O ESTADO DA PESQUISA

ESTADO	PROJ	ESTADO	PROJ	ESTADO	PROJ
AC	12	AL	23	AM	22
AP	7	BA	112	CE	45
DF	2	ES	52	GO	92
MA	10	MG	100	MT	109
PA	27	PB	75	PE	50
PI	52	PR	76	RD	2
RJ	103	RN	77	RD	1
RS	92	SC	82	SE	25
SP	240				
				TOTAL	1488

O pedido é para uma relação do número de projetos por estado. O sistema pesquisa o arquivo inteiro, sequencialmente, e constrói a resposta na memória à medida que encontra novos valores pivotais. A resposta é ordenada numa sequência ascendente dos valores pivotais e sempre inclui o total. A relação é repetida em várias colunas para utilizar ao máximo o espaço disponível para apresentação.

Comando: Selecione PROJ com estado 'C'
Conte PROJ por estado

Resposta: EFETUANDO UMA PESQUISA SEQUENCIAL NO ARQUIVO PROJ
ENTRE # PARA INTERROMPER E CANCELAR A PESQUISA.
ENTRE ? PARA SABER O ESTADO DA PESQUISA.

FICHAS PROJ SELECIONADAS: 176

ESTADO	PROJ	ESTADO	PROJ	ESTADO	PROJ
AC	12	AL	23	AM	22
AP	7	BA	112		
				TOTAL	176

Os verbos Conte e Totalize podem funcionar junto com o verbo Selecione. O verbo selecione limita o domínio de fichas consideradas pelos verbos Conte e Totalize.

Comando: Selecione PROJ com estado 'C'
Totalize valor global PROJ por estado

Resposta: EFETUANDO UMA PESQUISA SEQUENCIAL NO ARQUIVO PROJ

ENTRE # PARA INTERROMPER E CANCELAR A PESQUISA
ENTRE ? PARA SABER O ESTADO DA PESQUISA

FICHAS PROJ SELECIONADAS: 176

ESTADO	VALOR GLOBAL.PROJ	ESTADO	VALOR GLOBAL.PROJ
AC	92.000.000,00	AL	85.000.000,00
AM	130.970.000,00	AP	6.000.000,00
BA	827.722.500,00		
		TOTAL	1.141.692.500,00

Este comando mostra a relação e totalização dos valores globais de todos os projetos nos cinco estados indicados. Cada ficha proj tem um único valor no campo valor.global proj. O sistema percorre o arquivo inteiro; seleciona as fichas com um valor menor que C no campo estado, conta uma tabela dos estados selecionados, e de cada ficha selecionada, extrai o valor do campo valor.global.proj e soma no total mantido por estado. Quando chegar ao fim do arquivo então o total por estado é a soma dos valores globais dos projetos daquele estado, sejam os projetos dois ou duzentos. O sistema então formata a tabela, calcula o total geral e imprime.

Comando: TOTALIZE POR FONTE PROJ E TOTAL FONTE

Resposta: EFETUANDO UMA PESQUISA SEQUENCIAL NO ARQUIVO PROJ
ENTRE # PARA INTERROMPRE E CANCELAR A PESQUISA
ENTRE ? PARA SABER O ESTADO DA PESQUISA

FONTE.RECURSOS	VALOR.GLOBAL.PROJ	TOTAL.REC.FONTE
BNDE	2.722.845.000,00	1.500.000.000,00
CNPQ	715.000.000,00	600.000.000,00
FINEP	1.520.000.000,00	1.000.000.000,00
MINIPLAN	90.370.000,00	70.000.000,00
TOTAL	5.048.215.000,00	3.170.000.000,00

O nome do arquivo não consta neste comando, mas é derivado dos nomes dos campos. O sistema não permite que um nome de campo pertença a mais que um arquivo. A ausência do nome do arquivo implica o uso de uma lista orientadora se essa fosse criada pelo uso anterior do verbo Selecione.

Este comando demonstra a totalização de valores armazenados em fichas e em subfichas subordinadas às fichas. O campo sendo usado como campo pivotal de classificação é um campo da subficha. Um campo sendo totalizado é da subficha também (TOTAL.REC.FONTE) mas o outro campo sendo totalizado é da ficha logicamente acima da subficha. Essa estrutura pode dar um resultado sem sentido lógico. Considere a seguinte ficha e as suas subfichas relacionadas:

FICHA PROJ

PROJ: 77702
=
=
=
VALOR.GLOBAL.PROJ: 600.000,00
=
=
=

FONTE:	CNPQ
EQUIP:	=
INVES:	=
PESS :	=
CUST :	=
OUTRO:	=
TOTRF:	200.000,00

FONTE:	BNDE
EQUIP:	=
INVES:	=
PESS :	=
CUST :	=
OUTRO:	=
TOTRF:	300.000,00

FONTE:	FINEP
EQUIP:	=
INVES:	=
PESS :	=
CUST :	=
OUTRO:	=
TOTRF:	100.000,00

SUBFICHAS ALOC.RECURSOS (TOTAL.REC.FONTE=TOTRF)

O valor do campo VALOR.GLOBAL.PROJ para esse projeto corresponde à soma dos totais dos recursos alocados em três subfichas das fontes. Como o computador não pode saber intuitivamente, que o valor global do projeto é logicamente ligado ao TOTRF, ele não é capaz de dividir o valor 600.000,00 entre as três fontes. Portanto, o usuário pediu uma totalização em relação à fonte. O computador sempre assume que o usuário sabe o que está fazendo. A única decisão que ele pode fazer é repetir o valor global três vezes, uma vez por fonte. O total do valor.global.proj então com classificação por fonte será 1.800.000,00.

Realmente, o pedido de uma totalização do valor.global. proj por fonte não tem sentido. O sistema permite que o usuário peça essa totalização; cabe ao usuário analisar o que ele está pedindo. É por esta razão que o total apresentado no exemplo acima para valor.global.proj é maior que o total que seria dado pelo comando "Totalize valor.global.proj".

Comando: Selecione PROJ com setor '00100' até '00104' el com estado 'RJ' 'SP' 'MG'.
Totalize valor.global.proj por setor

Resposta: EFETUANDO UMA PESQUISA SEQUENCIAL NO ARQUIVO PROJ
ENTRE # PARA INTERROMPER E CANCELAR A PESQUISA
ENTRE ? PARA SABER O ESTADO DA PESQUISA

SETOR	VALOR.GLOBAL.PROJ
00100	27.420.200,00
00101	3.100.000,00
00103	100.000.000,00
00104	66.733.000,00
TOTAL	198.053.200,00

Este comando mostra um exemplo simples de uma seleção e uma totalização.

Comando: Selecione PROJ com setor '00090' até '00104' el com estado 'RJ' 'SP' 'MG'
Totalize valor.global.proj por setor '0090,5' .... '00100001041;2'

Resposta: EFETUANDO UMA PESQUISA SEQUENCIAL NO ARQUIVO PROJ
ENTRE # PARA INTERROMPER E CANCELAR A PESQUISA
ENTRE ? PARA SABER O ESTADO DA PESQUISA.

PROJ COM SETOR '00090' ATÉ '00104' EL COM ESTADO 'RJ' 'SP' 'MG'.

Comando: Imprima Num.LIB Data.Prev.LIB e Valr.Prev.LIB das LIBERAÇÕES DO PROJETO '1Ø27Ø'

Resposta:

NUM.LIB	DATA.PREV.LIB	VALR.PREV.LIB
01	01/10/77	200.000,00
02	01/12/77	200.000,00

Este comando ilustra uma limitação de impressão das subfichas LIBE da ficha PROJ '∅127∅'. Primeiro, o identificador da ficha não está impresso por causa da palavra chave DO na frente do nome do arquivo PROJ (funciona como SO mas com localização restrita). Segundo, os campos da subficha são limitados a três. Terceiro, o usuário especifica as subfichas a serem impressas. As subfichas não têm identificadores. Então o usuário seleciona as subfichas desejadas através da palavra chave COM junto com um valor entre apóstrofes e um indicador da seleção entre este valor e o valor armazenado nas subfichas a serem aceitas ($<$ é o indicador de "menor que" assim como $>$ é o indicador de "maior que"). Note que o sistema verificará a data de todas as liberações associadas ao projeto; e le não toma conhecimento de uma ordenação para deduzir que todas as subfichas além da primeira não aceita também serão fora do critério de seleção.

Comando: Selecione PROJ com estado 'RJ' 'SP' 'MG' 'MT'
Conte PROJ por setor '00100-00104,1'
Totalize valor.global.proj

Resposta: EFETUANDO UMA PESQUISA SEQUENCIAL NO ARQUIVO PROJ
ENTRE # PARA INTERROMPER E CANCELAR A PESQUISA
ENTRE ? PARA SABER O ESTADO DA PESQUISA

PROJ COM ESTADO 'RJ' 'SP' 'MG' 'MT'

SETOR	PROJ	VALOR.GLOBAL.PROJ
00100	5	34.500.200,00
00101	1	3.900.000,00
00102	0	0,00
00103	1	100.000.000,00
00104	3	72.733.000,00
OUTRO	1478	259.923.566.800,00
TOTAL		260.132.700.000,00

Este comando mostra uma combinação dos verbos conte e totalize. O campo pivotal (setor) está especificado uma só vez. O verbo totalize assume a mesma organização que o verbo conte. Este comando mostra também a especificação de faixas de classificação. Estão especificados cinco intervalos individu-

ais e um intervalo geral para todos valores não incluídos nos individuais.

Comando: Selecione PROJ com estado 'R' EL com setor.....
'00750' até '00752'
CONTE PROJ por estado e por setor

Resposta: EFETUANDO UMA PESQUISA SEQUENCIAL NO ARQUIVO PROJ
ENTRE # PARA INTERROMPER E CANCELAR A PESQUISA
ENTRE ? PARA SABER O ESTADO DA PESQUISA

PROJ COM ESTADO 'R' EL COM SETOR '00750' ATÉ ....
'00752'

ESTADO	SETOR			
	00750	00751	00752	TOTAL
RD	0	0	1	1
RJ	2	0	0	2
RN	7	5	3	15
RO	0	0	1	1
RS	0	2	0	2
SC	0	3	0	3
SE	0	5	2	7
SP	1	0	0	1
TOTAL	10	15	7	32

Capítulo V - DESCRIÇÃO ANALÍTICA DA TABULAÇÃO

O módulo de tabulação é chamado pelo pré-processa - dor se identificar os verbos CONTE ou TOTALIZE. A seguir são realizadas 3 etapas de montagem de tabelas e logo após o controle é retornado ao pré-processador. Ele então chama a etapa 4 ou de execução propriamente dita da tabulação. Logo após, a etapa 5 prepara os dados tabulados para o módulo de impressão imprimir os resultados finais.

ETAPAS DE MONTAGEM

ETAPA 1

A etapa 1 da tabulação analisa o texto do pedido do usuário. Dependendo do tipo do verbo (Conte ou Totalize) arma flags diferentes para a posterior identificação do verbo. A seguir, coloca o campo correspondente ao verbo, como um elemento na Tabela I(tabela de impressão, ilustrada no apêndice A) indicando na mesma, o flag correspondente ao verbo. Os outros campos do elemento são preenchidos na mesma tabela, com base no dicionário de campos. O dicionário de campos (DICI) é um arquivo mestre TOTAL que contém todas as especificações de cada campo como nome usuário do campo, nome total, tamanho de armazenamento, formato de armazenamento e outros campos. O dicionário de campos é ilustrado no apêndice B.

Caso o campo correspondente ao verbo não seja encontrado na Tabela I, a etapa 1 desvia para uma rotina de erro padrão do STELA que indica o tipo do erro e caso a modalidade de processamento não seja "batch", permite a reentrada do nome do campo correto através do terminal.

Nesta busca os espaços em branco e o separador "E" são ignorados. Os outros campos encontrados no texto são interpretados como pivôs. Neste caso, é armado um flag diferente para pivô. Da mesma maneira, as informações correspondentes dos campos pivôs são colocadas como elementos na Tabela I com base no dicionário de campos.

Da mesma forma, caso o campo não seja encontrado, existe o desvio para as rotinas padrões de erro do sistema.

Quando é alcançado o fim de texto (#) o controle é passado ao pré-processador que transfere controle para etapa 2.

As macros STELA usadas na confecção da etapa 1 são:

- TEST PALV - verifica se a palavra encontrada no texto é idêntica à passada como parâmetro.
- PROX PALV - localiza a próxima palavra do texto
- PROC NOME - pesquisa a ocorrência do pivô no dicionário de campos e no caso de ser encontrado cria o elemento correspondente ao pivô na Tabela I.
- CALC ERRO - trata as diferentes condições de erro

ETAPA 2

A etapa 2 constrói as listas de elementos do TOTAL na área de dados do usuário (11). É construída com base na tabela I e completada com dados da tabela R, que é a tabela que descreve os arquivos. Esta tabela é melhor explicada no apêndice C.

Os dados da tabela R usados são: o número do arquivo na hierarquia do fluxograma geral, a localização da área de leitura, o tamanho da área de leitura, o tamanho do esquema de projeção e quando necessário, o indicador do caminho de ligação entre o arquivo principal e um arquivo tipo variável.

As localizações dos campos a serem lidos são determinadas em relação ao começo das respectivas áreas de leitura e são usadas para atualizar as tabelas I e C.

A montagem das listas de elementos tem de esperar a conclusão da primeira etapa devido aos arquivos associados e aos domínios com modificadores funcionais. A resolução das localizações dos campos associados só pode ser feita depois de considerar-se todos os campos principais; um campo usado em funções diferentes pode aparecer em dois esquemas de projeção diferentes. Portanto, esses não podem ser montados juntamente com o reconhecimento dos campos na etapa 1. A etapa dois colo

ca uma instrução de leitura na tabela C para cada lista de elementos montada.

As principais macros STELA usadas na etapa 2 são:

- TERM TAB - cria elemento final na tabela indicada por um dos parâmetros.
- ELEMLIST - gera as listas de elementos TOTAL a serem usadas na etapa 4.
- RESLVLOC - calcula a localização dos elementos na IO AREA TOTAL.
- RESTRTAB - posiciona o primeiro elemento da tabela indicada por um dos parâmetros.

ETAPA 3

Esta etapa é a mais simples de todas. Ela consiste, basicamente, em selecionar alguns campos dos elementos da tabela I e transferi-los para uma outra tabela chamada PIVTAB . Esta tabela contém um resumo da tabela I pois somente alguns campos estão ali copiados, bem como possui ordenação diferente da mesma. Esta tabela encontra-se melhor descrita no apêndice D.

Os campos de PIVTAB servirão de parâmetros para a execução propriamente dita da tabulação na etapa 4, bem como para a preparação da formatação de impressão na etapa 5.

Além disso, são acumulados para fins de controle , também na execução, o número total de pivôs em NPIV e o número total de verbos conte e totalize em NTOT.

A tabela PIVTAB é carregada através de sucessivas leituras sequenciais da tabela I e é ordenada colocando-se inicialmente os elementos pivôs, depois os elementos correspondentes aos verbos conte e finalmente os correspondentes ao verbo totalize. Além disso para um mesmo tipo de elemento é respeitada a ordem de entrada do comando. A condição para a identificação do elemento é o flag armado ainda na etapa 1.

Após o funcionamento da etapa 3 todas as tabelas necessárias à execução propriamente dita da tabulação estão perfeitamente montadas e o controle é então passado ao pré-pro -

cessador para permitir condição futura de integração entre os vários módulos do STELA.

O pré-processador aciona então a etapa 4.

As principais macros STELA usadas na etapa 3 são:

- RESTRTAB - já usada na etapa 2
- NXTINTAB - posiciona o elemento seguinte na tabela indicada por um dos parâmetros da macro.

ETAPA 4

Consiste da tabulação propriamente dita. Nesta etapa são usadas as listas de elementos para o TOTAL formadas na etapa 2 e que se encontram na página correspondente à área de dados de ordenação. Além disso serão usadas as informações criadas na etapa 2 com base na tabela R e que indicam a localização (deslocamento) da lista de elementos bem como o deslocamento da IOAREA na qual os dados recuperados do banco de dados são colocados.

A IOAREA funciona como um buffer de recuperação de dados. Também os esquemas de inicialização dos arquivos TOTAL bem como os esquemas de leitura de mestre e variáveis criados no final da etapa 2, são utilizados e passados como parâmetros nas macros STELA que executam as funções de inicialização e leitura dos arquivos.

Os dados da tabulação são colocados numa página correspondente à área A3 (também de 4 k) e tratamento especial de estouro da área é previsto quando o volume de tabulação ameaçar ultrapassar este espaço. Esta área é dividida em registros de tamanho variável pois o tamanho do conteúdo de cada valor de pivô é diferente. Nenhum espaço entre registros é perdido, para possibilitar aproveitamento integral da memória. Os registros são preenchidos na área sequencialmente, à proporção que são recuperados do banco de dados. Posteriormente, condições de restrições poderão ser introduzidas para permitir tabular apenas determinados intervalos.

Os algorítmos de tabulação estão elaborados de tal

maneira que comportam qualquer nível de pivôs ou seja: a hierarquia de contagem ou totalização poderá ter um número ilimitado de níveis, não obstante as condições práticas de uso raramente necessitarem de mais de três níveis.

Exemplificando melhor, quando usamos: CONTE PROJETO POR ESTADO E POR SETOR # estamos trabalhando com dois níveis de pivôs. O 1º nível na hierarquia é ESTADO e o outro SETOR. Contaremos então o número de projetos em cada estado num 1º nível e dentro de cada estado, por setor, num 2º nível hierárquico.

A representação desta estrutura hierárquica ou em árvore, na memória, foi detidamente pesquisada. Chegamos à conclusão que a melhor forma de representação seria através de uma lista encadeada usando-se dois tipos de ponteiros. O 1º também chamado de irmão, apontará sempre para o registro seguinte de mesmo nível de pivô. Por exemplo AM (Amazonas) apontará para BA (Bahia) e este para CE (Ceará). O 2º ponteiro, também chamado de filho, apontará sempre para o elemento de nível hierárquico imediatamente abaixo. Exemplificando, o elemento CE (Ceará) poderá ter o ponteiro do filho apontando para o elemento do setor 4.000 que por sua vez terá o ponteiro do irmão indicando o setor 5.000 (também de Ceará). O elemento correspondente ao último nível na hierarquia, conterá então o contador que indicará o total de ocorrências encontradas na busca e/ou os totalizadores (no caso do verbo totalize) com os valores acumulados dos campos a serem totalizados, também no último nível da hierarquia.

Exemplificando melhor:

ESTADO	PROJ	ESTADO	PROJ	ESTADO	PROJ
AC	12	AL	22	AM	22
AP	7	BA	112	CE	45
DF	2	ES	52	GO	92
MA	10	MG	100	MT	109
PA	27	PB	75	PE	50
PI	52	PR	76	RD	2
RJ	103	RN	77	RO	1
RS	92	SC	82	SE	25
SP	240			TOTAL	1488

Conte projeto por estado e por setor

Registros do Banco de Dados na ordem da recuperação

ESTADO	SETOR
RJ	6000
SP	3400
AM	3400
RJ	6000
RJ	3400
RJ	4000

Representação da tabela na memória após a tabulação

Endereço	Link mesmo nível	Link nível inferior	Tamanho	Identif	Conteúdo	Contador
1	3	7	2		RJ	
2	ø	ø	4		6000	00002
3	ø	4	2		SP	
4	ø	ø	4		3400	00001
5	1	6	2		AM	
6	ø	ø	4		3400	00001
7	8	ø	4		3400	00001
8	2	ø	4		4000	00001

Para o estabelecimento da operação de encadeamento dos registros, é realizada para cada registro recuperado do banco, uma busca na área de tabulação, seguindo-se os ponteiros para a determinação da "localização lógica de inserção". Caso o elemento seja encontrado em todos os seus níveis, são apenas acumulados os contadores e/ou totalizadores. Caso ainda não exista a ocorrência para todos ou alguns níveis de pivô, somente os registros para os níveis inexistentes são criados, respeitando-se porém o encadeamento lógico.

Uma melhoria introduzida para permitir maior facilidade de percorrer a árvore na etapa seguinte é através do uso do ponteiro de irmão para outra finalidade. Quando não houver mais ocorrência de irmão, em qualquer nível, ao invés de se usar o ponteiro nulo, aproveita-se o mesmo para apontar para o elemento pertencente ao pai. Isto permite melhor capacidade de subir na estrutura sem a necessidade do uso de pilhas que seriam neste caso de elaboração mais trabalhosa e possivelmente menos eficiente.

No encadeamento dos elementos é usado um ponteiro de apenas 2 bytes (uma halfword) que contém apenas o deslocamento em hexadecimal para o registro que está sendo apontado. Este deslocamento é sempre em relação ao início da área de trabalho, endereço esse, contido no registrador base da área. Aliás para todos os encadeamentos internos em memória, no STELA, são usados apontadores de 2 bytes, enquanto os ponteiros para as áreas intermediárias de disco, tem o tamanho de 4 bytes (uma fullword). Isto porque deve permitir registrar o TTR correspondente ao registro no disco. O TTR permite localizar o endereço relativo do registro na trilha e cilindro correspondente no disco.

A seguir examinaremos detalhadamente o formato do registro nesta área de trabalho.

Os campos componentes são os seguintes:

- apontador para o 1º irmão - tamanho de 2 bytes
- apontador para o 1º filho - tamanho de 2 bytes
- identificador do registro - tamanho de 1 byte
- tamanho do valor - tamanho de 1 byte

- espaço para uso futuro - tamanho de 2 bytes

Esta parte do conteúdo de todo registro é de tamanho fixo.

- conteúdo do valor - tamanho variável
- contador para o verbo conte- tamanho de 4 bytes
- totalizadores - tamanho de 8 bytes

Consequentemente este trecho do registro é de tamanho variável, pois depende do tamanho do valor do pivô, bem como dos contadores e/ou totalizadores utilizados.

As macros STELA utilizadas na etapa 4 são:

- INIREAD - open nos arquivos TOTAL
- READMES - leitura de arquivos mestres TOTAL

ETAPA 5

Esta etapa é a responsável pela formatação dos dados para serem exibidos ao usuário. O módulo responsável pela exibição é o de impressão. Esta etapa grava em disco (STELWORK), a lista orientadora que servirá de base para a impressão. São gravados em disco, blocos de 512 bytes encadeados pelo TTR. O 1º bloco contém o cabeçalho da lista orientadora que indica os campos (pivôs), contadores e totalizadores a serem impressos. Este bloco é gravado através de uma leitura sequencial na PIVTAB. Posteriormente os outros blocos são gravados com base na lista encadeada de tabulação criada na etapa 4. Esta lista é percorrida seguindo-se os ponteiros de irmão e filho, descendo-se na estrutura até o mais baixo nível de pivô e contador/totalizador e depois subindo novamente até que no 1º nível de pivô seja encontrado um ponteiro de irmão com marca de final.

A etapa 5 emite mensagem indicando o nome dos campos que foram tabulados bem como o nome dos contadores e totalizadores que servirão para compor a linha de cabeçalho da impressão

Interfaces do módulo de tabulação com os módulos de controle do STELA

O módulo de controle reside no núcleo do STELA e é responsável pelo controle centralizado das operações de entrada/saída, além da alocação dos recursos de máquina a cada tarefa responsável pela execução dos comandos (2). O módulo de controle é dividido em 2 partes.

1- Supervisão de tarefas

2- Supervisão de entrada/saída

Uma das funções mais importantes do módulo de controle, que o módulo de tabulação usa, são as transferências de controle, que são executadas entre módulos paginávels ou não. Por exemplo, a ligação normal entre a 1ª etapa de tabulação e a 2ª etapa, no caso de não ter sido encontrada nenhuma condição de anormalidade do texto do pedido e de todos os campos terem sido criados como elementos na tabela I, é através de uma rotina de TC (Transferência de Controle).

Estas rotinas cuidam da passagem de parâmetros e determinam os requisitos que as páginas de trabalho de cada módulo, necessitarão para funcionar, além de funções adicionais como salvamento e restauração de registradores.

Para isto são usadas as macros do STELA, TC (transfira controle, TCR (transfira controle com retorno) e RC (retorne controle).

Caso a etapa 1 termine normalmente, é dado o comando: TC TABØ2Ø, V que significa transferência de controle para a página aonde está contida a 2ª etapa (TABØ2Ø). O parâmetro "V" indica que este módulo é virtual ou seja, caso a página não esteja na memória deverá ser recuperada em disco e transferida para a memória.

A principal rotina de TC é a T#CNTL. Esta rotina, realiza as funções principais necessárias à transferência de controle, sendo que todas as demais são subrotinas desta. Descobre qual a macro chamada (TC, TCR ou RC), identifica a página que contém o próximo módulo e determina quais são as suas necessidades em termos de áreas de trabalho. Além disso, é feito um levantamento de quantas posições de memória da pilha de transferência de controle (TCSTACK) são necessárias para o armazenamento dos dados do módulo anterior, dos parâmetros e registradores. A pilha então é atualizada e as indicações ne-

cessárias para o próximo módulo são armazenadas na UCB (bloco de controle do usuário) e o controle é passado ao despachante.

Se a etapa 1 da tabulação encontra uma condição de erro não recuperável, desvia para uma rotina em que é usada a macro RC. Neste caso, o módulo de tabulação é descontinuado e o controle retorna definitivamente para o pré-processador.

A transferência de controle da etapa 2 para a etapa 3 é feita também através da macro TC; enquanto na ligação entre a etapa 3 (última da fase de montagem de tabelas) e a etapa 4 (execução) é usada a macro RC. Isto para permitir no futuro uma maior possibilidade de interligação entre as várias funções do STELA como seleção, ordenação, cálculo e impressão. As outras subrotinas usadas no processo de transferência de controle são:

1- TCFINDP - para a localização de um elemento na tabela de páginas.

2- TCFINDI - para a localização do elemento que descreve o próximo módulo à receber controle.

3- TC2 - esta subrotina procura os elementos livres da pilha de TC para determinar o próximo livre que possui tamanho N bytes. Se encontrar, aloca estes bytes, em caso contrário, chama principalmente a subrotina de limpeza de lixo.

4- TC3 - libera um bloco da pilha.

5- GARBAGE - reune todas as áreas livres ou abandonadas de TCSTACK. Determina os blocos livres ou abandonados de TCSTACK e os reune através de uma compressão das áreas ocupadas. Todos os ponteiros são atualizados e o resultado final é uma única área livre de tamanho máximo.

Na etapa 4 é usada a macro TCR em 2 casos. Inicialmente para passar o controle para a rotina de abertura dos arquivos TOTAL e posteriormente para a rotina de leitura de ar-

quivos mestre, TOTAL.

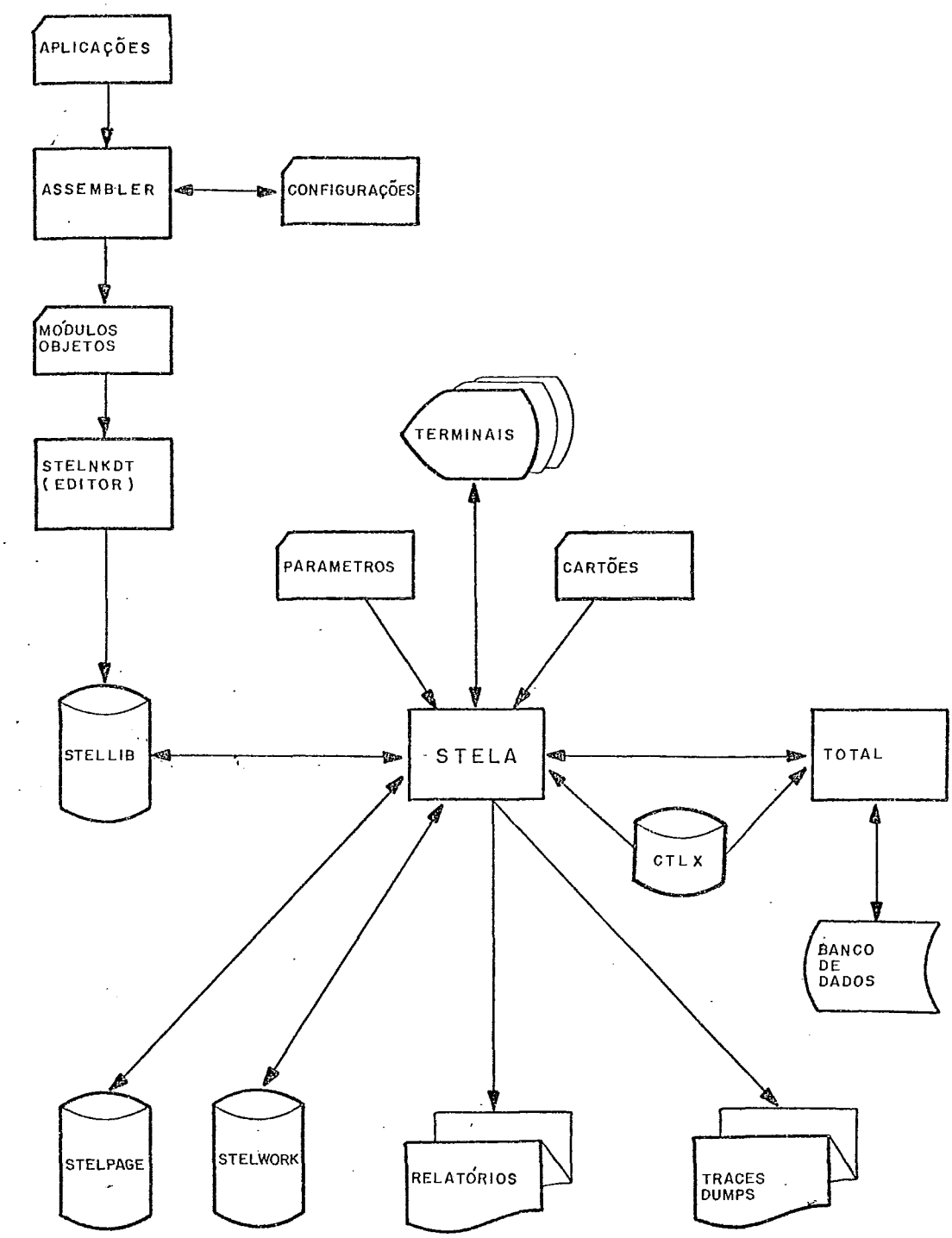
Nesta modalidade, o controle é passado, mas após o término da execução da rotina, ele é retornado para um ponto da etapa 4. Este ponto, no caso da leitura do arquivo mestre, pode variar, dependendo de ser encontrada a condição de fim de arquivo.

Capítulo VI º LÓGICA INTERNA DAS ETAPAS DA TABULAÇÃO

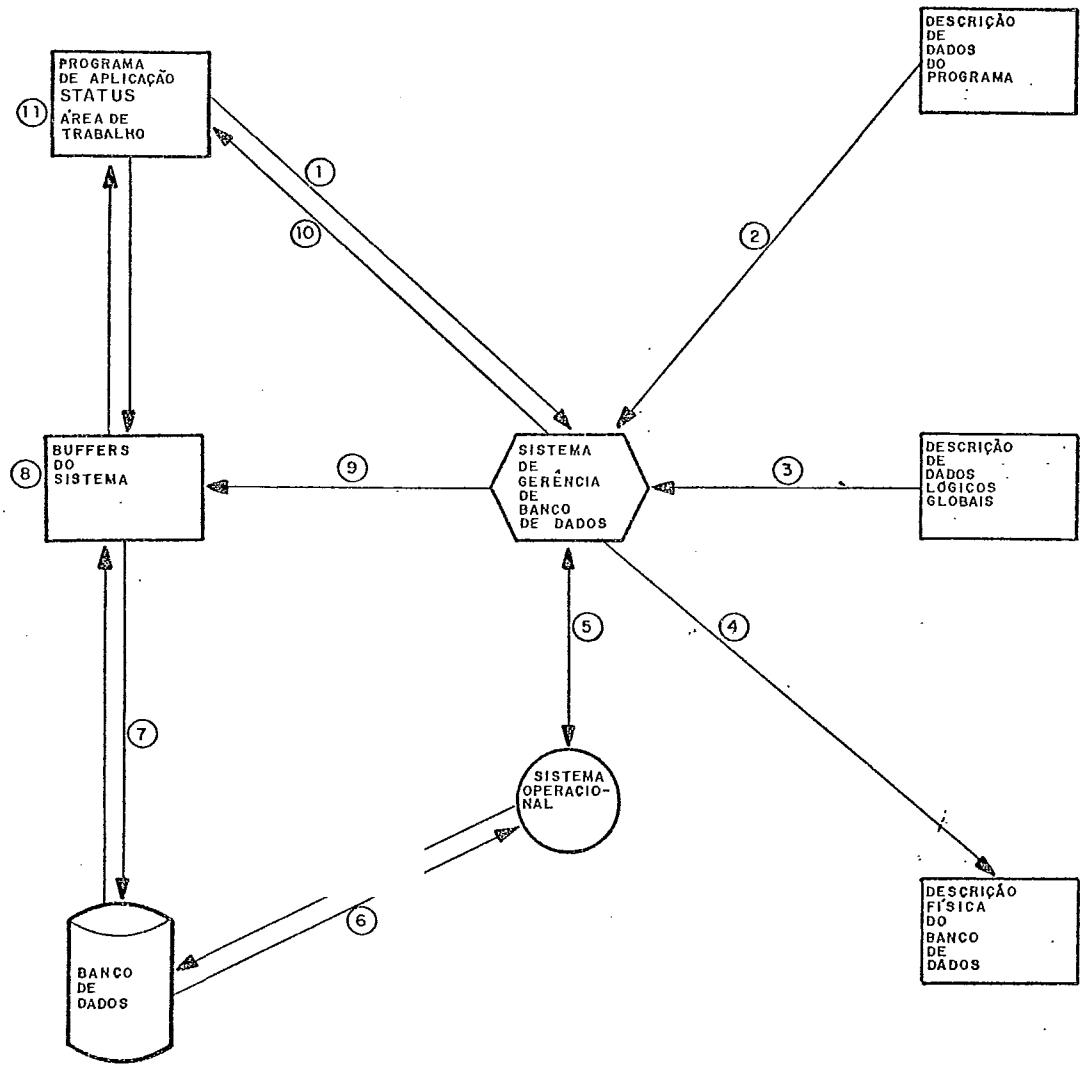
Para melhor compreensão da lógica, descrevemos a seguir, através de fluxos, os procedimentos de cada etapa.

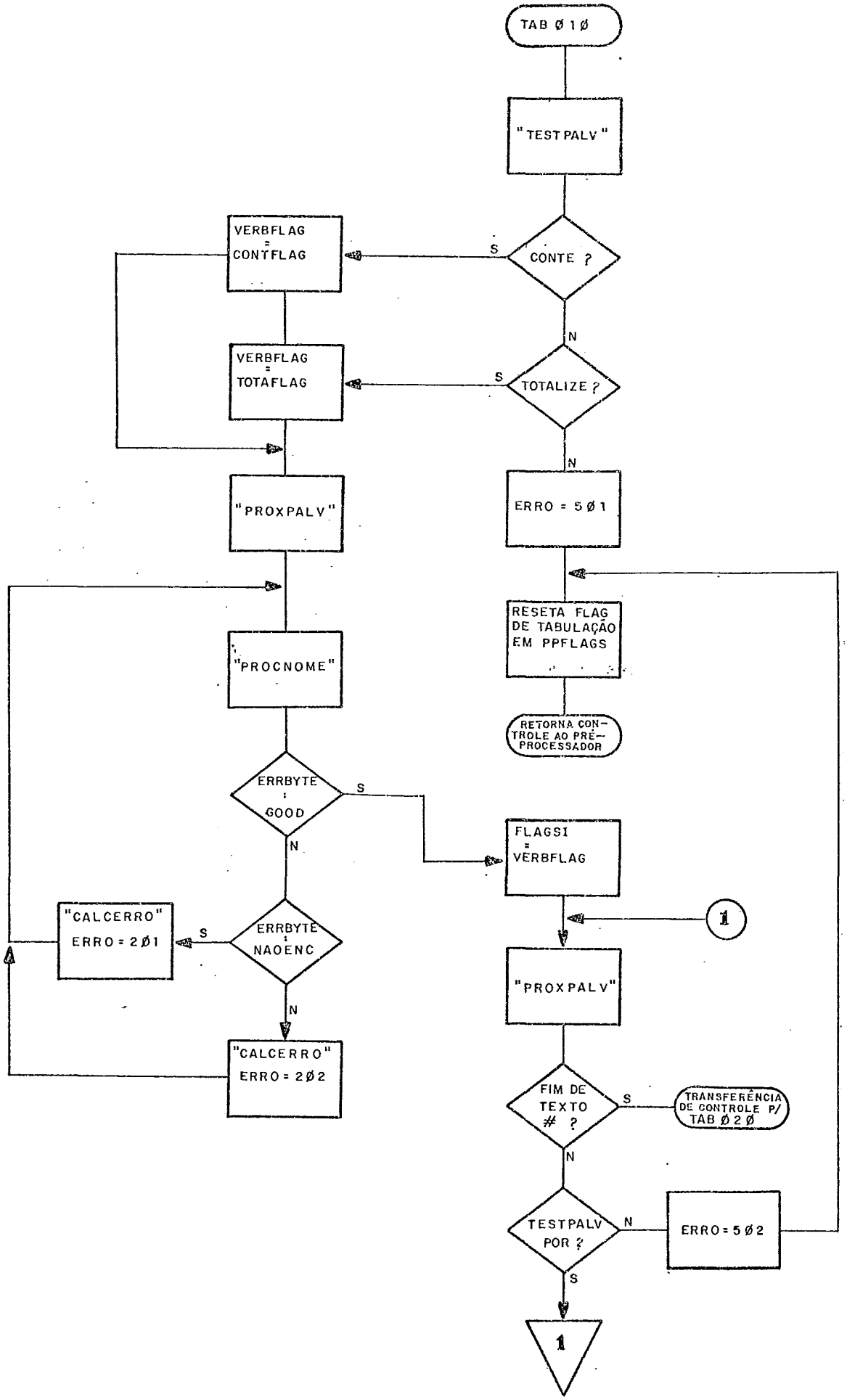
PROCESSO TRADICIONAL DE BUSCA

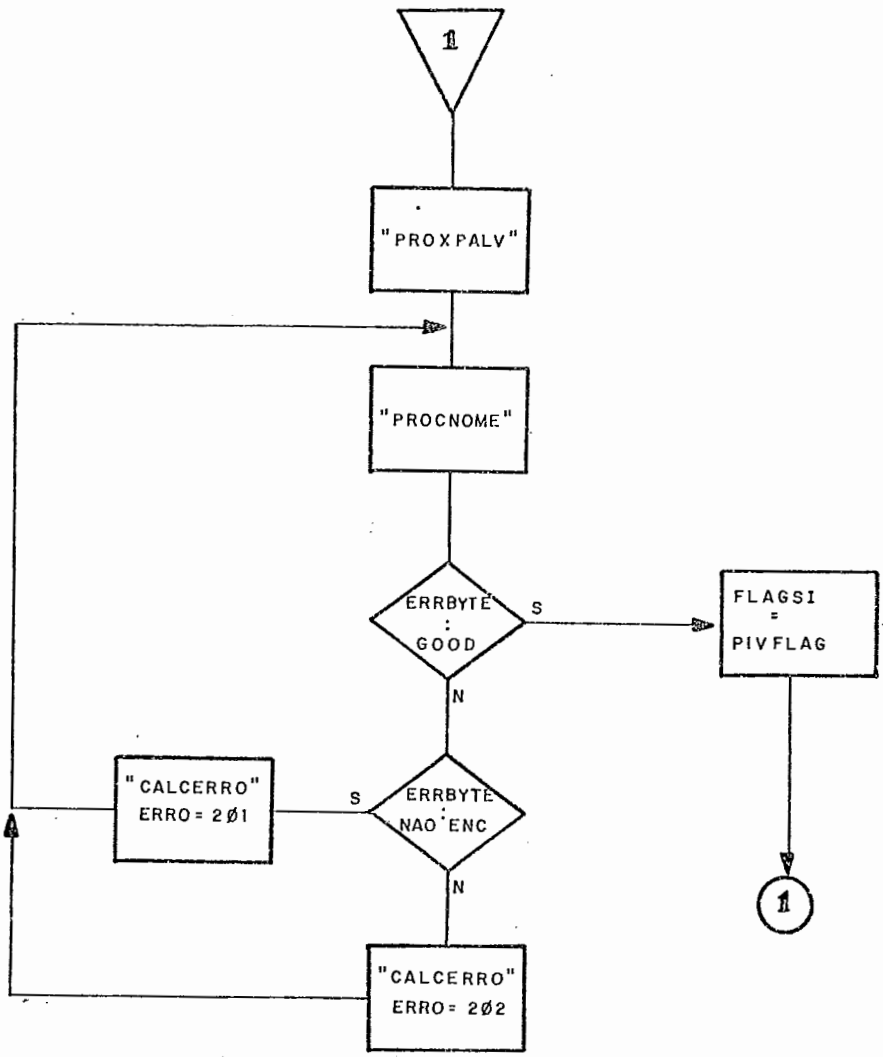
PROCESSO ATRAVÉS STELA

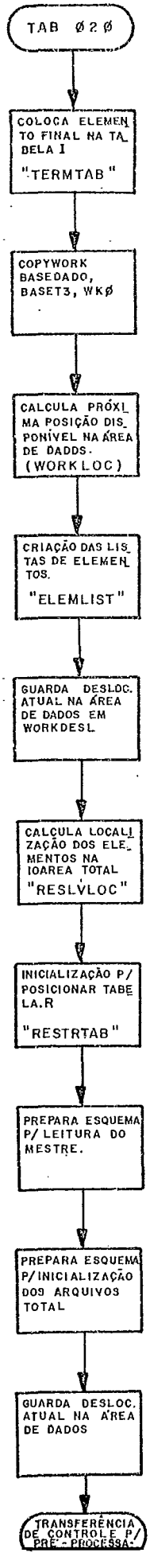


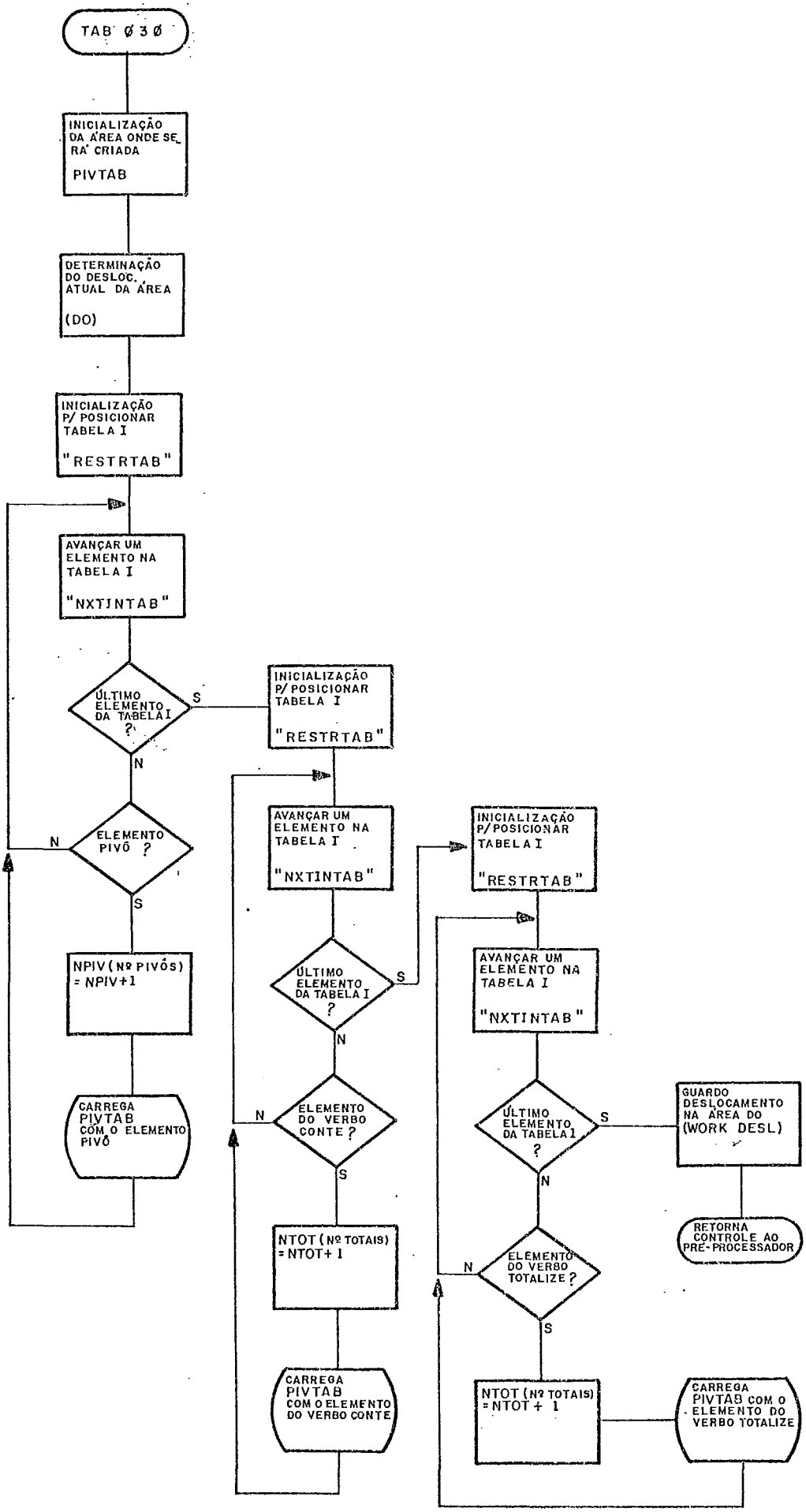
STELA-B : FLUXO GERAL DO SISTEMA

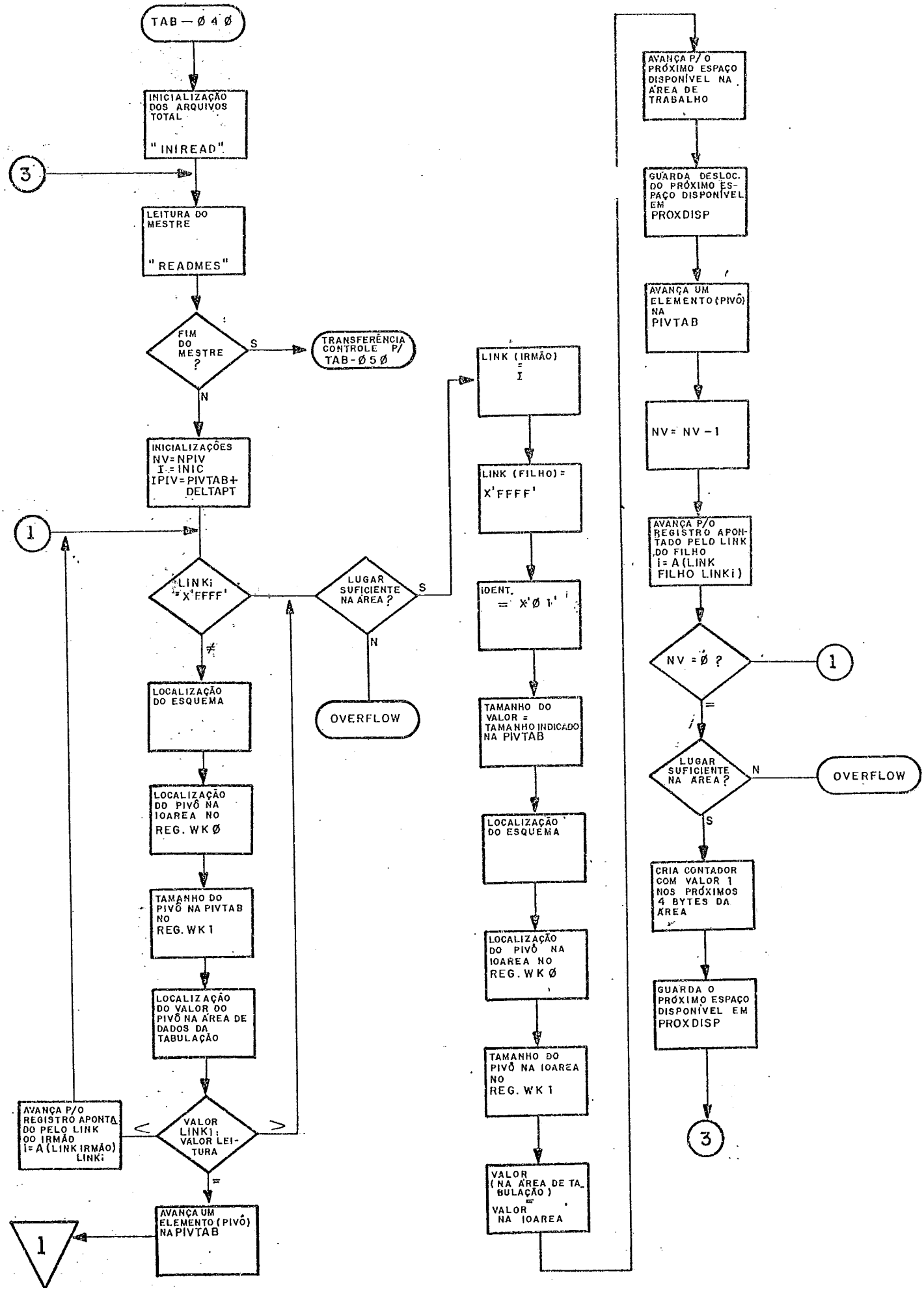

PROGRAMA DE APLICAÇÃO
STATUS
ÁREA DE TRABALHO
11
1
10
DESCRIÇÃO DE DADOS DO PROGRAMA
2
BUFFERS DO SISTEMA
8
9
SISTEMA DE GERÊNCIA DE BANCO DE DADOS
3
DESCRIÇÃO DE DADOS LÓGICOS GLOBAIS
5
4
7
SISTEMA OPERACIONAL
6
BANCO DE DADOS
DESCRIÇÃO FÍSICA DO BANCO DE DADOS


TAB Ø 1 Ø
"TESTPALV"
CONTE ?
S
VERBFLAG = CONTFLAG
N
TOTALIZE ?
S
VERBFLAG = TOTAFLAG
N
ERRO = 5Ø1
"PROXPALV"
RESETA FLAG DE TABULAÇÃO EM PPFLAGS
RETORNA CONTROLE AO PRÉ-PROCESSADOR
"PROCNOME"
ERRBYTE : GOOD
S
FLAGSI = VERBFLAG
N
1
"CALCERRO" ERRO = 2Ø1
S
ERRBYTE : NAOENC
N
"PROXPALV"
"CALCERRO" ERRO = 2Ø2
FIM DE TEXTO # ?
S
TRANSFERÊNCIA DE CONTROLE P/ TAB Ø 2 Ø
N
TESTPALV POR ?
N
ERRO = 5Ø2
S
1

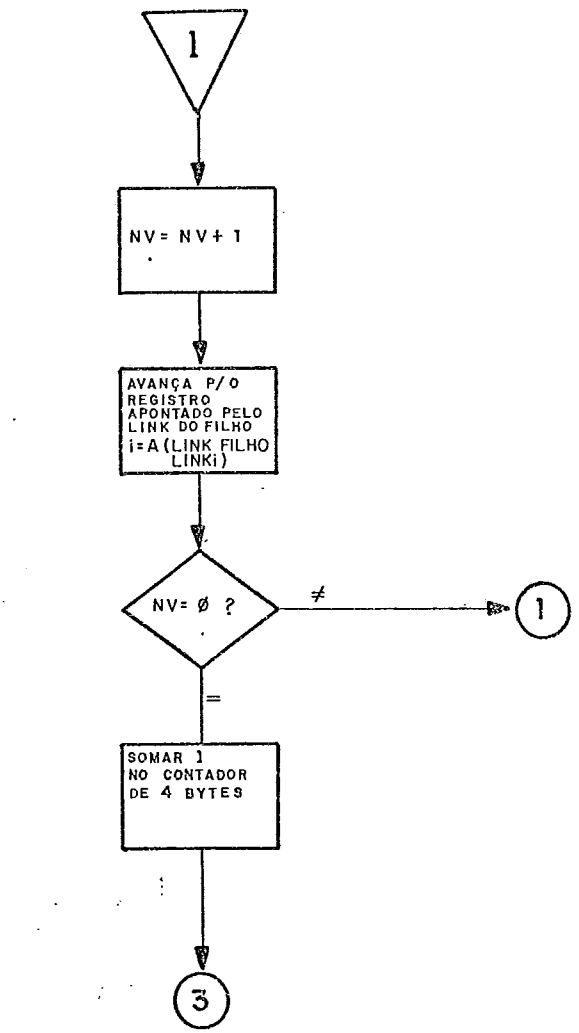

1
"PROXPALV"
"PROCNOME"
ERRBYTE : GOOD
S
FLAGSI = PIVFLAG
N
"CALCERRO" ERRO = 201
S
ERRBYTE : NAO ENC
N
1
"CALCERRO" ERRO = 202

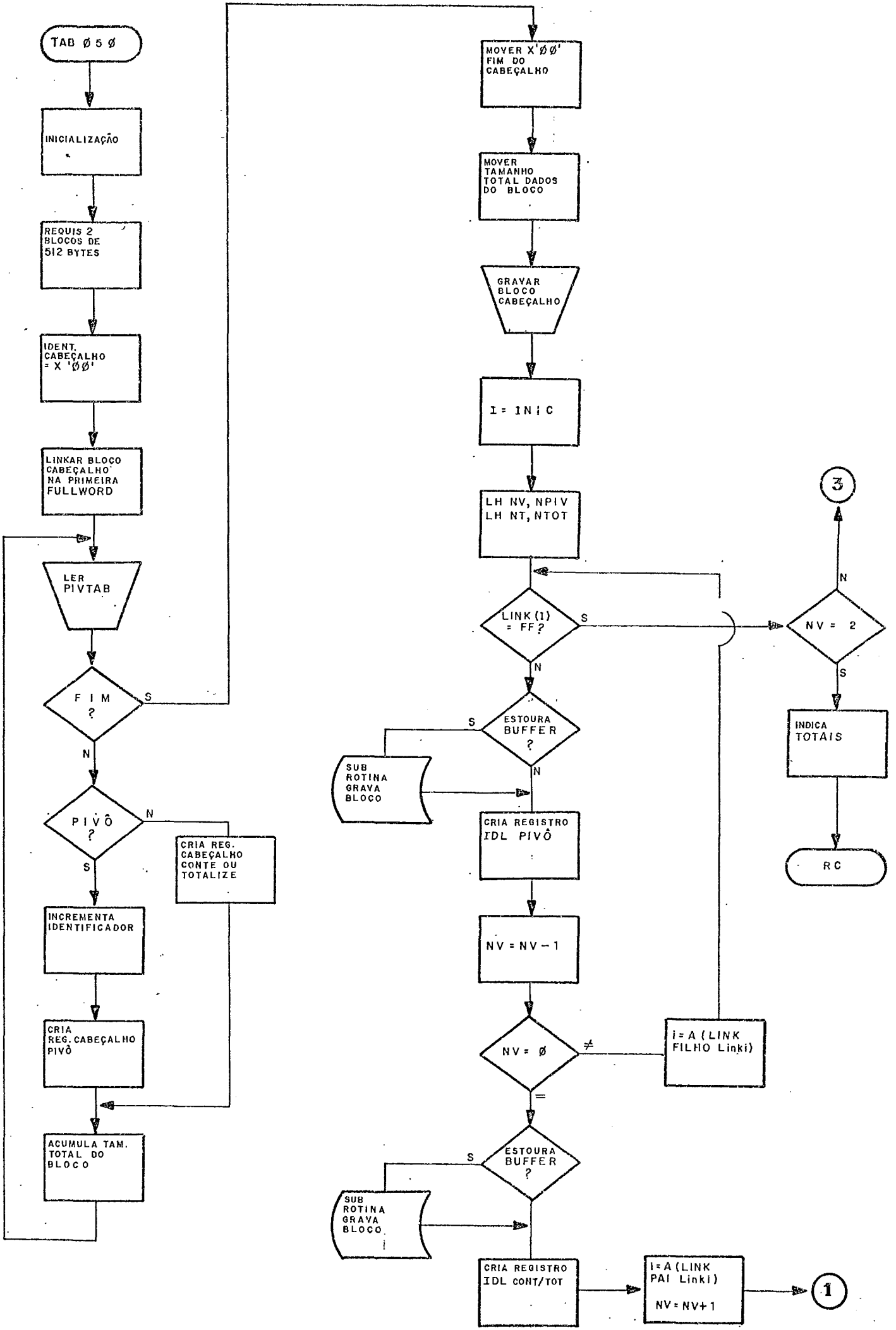

TAB Ø2Ø
COLOCA ELEMENTO FINAL NA TABELA I "TERMTAB"
COPYWORK BASEDADO, BASET3, WKØ
CALCULA PRÓXIMA POSIÇÃO DISPONÍVEL NA ÁREA DE DADDS. (WORKLOC)
CRIAÇÃO DAS LISTAS DE ELEMENTOS. "ELEMLIST"
GUARDA DESLOC. ATUAL NA ÁREA DE DADOS EM WORKDESL
CALCULA LOCALIZAÇÃO DOS ELEMENTOS NA IOAREA TOTAL "RESLVLOC"
INICIALIZAÇÃO P/ POSICIONAR TABELA.R "RESTRTAB"
PREPARA ESQUEMA P/ LEITURA DO MESTRE.
PREPARA ESQUEMA P/INICIALIZAÇÃO DOS ARQUIVOS TOTAL
GUARDA DESLOC. ATUAL NA ÁREA DE DADOS
TRANSFERÊNCIA DE CONTROLE P/ PRE - PROCESSA.


TAB Ø3Ø
INICIALIZAÇÃO DA ÁREA ONDE SERÁ CRIADA PIVTAB
DETERMINAÇÃO DO DESLOC. ATUAL DA ÁREA (DO)
INICIALIZAÇÃO P/ POSICIONAR TABELA I "RESTRTAB"
AVANÇAR UM ELEMENTO NA TABELA I "NXTINTAB"
ÚLTIMO ELEMENTO DA TABELA I ?
S
N
INICIALIZAÇÃO P/ POSICIONAR TABELA I "RESTRTAB"
ELEMENTO PIVÔ ?
N
S
NPIV (Nº PIVÔS) = NPIV+1
CARREGA PIVTAB COM O ELEMENTO PIVÔ
AVANÇAR UM ELEMENTO NA TABELA I "NXTINTAB"
ÚLTIMO ELEMENTO DA TABELA I ?
S
N
ELEMENTO DO VERBO CONTE ?
N
S
NTOT (Nº TOTAIS) = NTOT+ 1
CARREGA PIVTAB COM O ELEMENTO DO VERBO CONTE
INICIALIZAÇÃO P/ POSICIONAR TABELA I "RESTRTAB"
AVANÇAR UM ELEMENTO NA TABELA I "NXTINTAB"
ÚLTIMO ELEMENTO DA TABELA I ?
S
N
GUARDO DESLOCAMENTO NA ÁREA DO (WORK DESL)
RETORNA CONTROLE AO PRÉ-PROCESSADOR
ELEMENTO DO VERBO TOTALIZE ?
N
S
NTOT (Nº TOTAIS) = NTOT + 1
CARREGA PIVTAB COM O ELEMENTO DO VERBO TOTALIZE


TAB — Ø 4 Ø
INICIALIZAÇÃO DOS ARQUIVOS TOTAL
" INIREAD"
3
LEITURA DO MESTRE
" READMES"
FIM DO MESTRE ?
S
TRANSFERÊNCIA CONTROLE P/ TAB-Ø5Ø
N
INICIALIZAÇÕES
NV=NPIV
I=INIC
IPIV=PIVTAB+ DELTAPT
1
LINKi = X'EFFF'
≠
LUGAR SUFICIENTE NA ÁREA ?
S
N
OVERFLOW
LINK (IRMÃO) = I
LINK (FILHO) = X'FFFF'
IDENT. = X'Ø1'
TAMANHO DO VALOR = TAMANHO INDICADO NA PIVTAB
LOCALIZAÇÃO DO ESQUEMA
LOCALIZAÇÃO DO PIVÔ NA IOAREA NO REG. WKØ
TAMANHO DO PIVÔ NA IOAREA NO REG. WK1
VALOR (NA ÁREA DE TABULAÇÃO) = VALOR NA IOAREA
LOCALIZAÇÃO DO ESQUEMA
LOCALIZAÇÃO DO PIVÔ NA PIVTAB NO REG. WK1
LOCALIZAÇÃO DO VALOR DO PIVÔ NA ÁREA DE DADOS DA TABULAÇÃO
VALOR LINKi : VALOR LEITURA
<
>
=
AVANÇA P/O REGISTRO APONTADO PELO LINK DO IRMÃO I=A(LINK IRMÃO) LINKi
AVANÇA UM ELEMENTO (PIVÔ) NA PIVTAB
1
AVANÇA P/O PRÓXIMO ESPAÇO DISPONÍVEL NA ÁREA DE TRABALHO
GUARDA DESLOC. DO PRÓXIMO ESPAÇO DISPONÍVEL EM PROXDISP
AVANÇA UM ELEMENTO (PIVÔ) NA PIVTAB
NV = NV - 1
AVANÇA P/O REGISTRO APONTADO PELO LINK DO FILHO I=A(LINK FILHO LINKi)
NV = Ø ?
1
=
LUGAR SUFICIENTE NA ÁREA ?
N
OVERFLOW
S
CRIA CONTADOR COM VALOR 1 NOS PRÓXIMOS 4 BYTES DA ÁREA
GUARDA O PRÓXIMO ESPAÇO DISPONÍVEL EM PROXDISP
3

ETAPA - 4 /2


1
NV = NV + 1
AVANÇA P/ O REGISTRO APONTADO PELO LINK DO FILHO i = A (LINK FILHO LINKi)
NV = Ø ?
≠
1
=
SOMAR 1 NO CONTADOR DE 4 BYTES
3


TAB Ø5Ø
INICIALIZAÇÃO
REQUIS 2 BLOCOS DE 512 BYTES
IDENT. CABEÇALHO = X 'ØØ'
LINKAR BLOCO CABEÇALHO NA PRIMEIRA FULLWORD
LER PIVTAB
F I M ?
S
N
P I V Ô ?
N
S
CRIA REG. CABEÇALHO CONTE OU TOTALIZE
INCREMENTA IDENTIFICADOR
CRIA REG. CABEÇALHO PIVÔ
ACUMULA TAM. TOTAL DO BLOCO
MOVER X'ØØ' FIM DO CABEÇALHO
MOVER TAMANHO TOTAL DADOS DO BLOCO
GRAVAR BLOCO CABEÇALHO
I = IN I C
LH NV, NPIV
LH NT, NTOT
LINK (1) = FF?
S
N
ESTOURA BUFFER ?
S
N
SUB ROTINA GRAVA BLOCO
CRIA REGISTRO IDL PIVÔ
NV = NV – 1
NV = Ø
≠
=
I = A (LINK FILHO Link1)
ESTOURA BUFFER ?
S
SUB ROTINA GRAVA BLOCO
CRIA REGISTRO IDL CONT/TOT
I = A (LINK PAI Link1)
NV = NV+1
1
3
N
NV = 2
S
INDICA TOTAIS
RC

Capítulo VII - CONCLUSÕES FINAIS

Ainda não existe no Brasil uma metodologia de elaboração de software de base. Em países mais adiantados nesta área, como na Alemanha por exemplo, já estão sendo feitas tentativas de metodização. Constatou-se também que a maior dificuldade na elaboração de software básico, comparando-o com o de aplicação, é inerente ao maior grau de generalização e abstração necessárias na confecção do primeiro.

Pareceu-nos também que as linguagens tipo Assembler podem ser mais aconselháveis para este tipo de trabalho, que as de alto nível, devido ao grande número de decisões lógicas envolvidas. Por outro lado, observamos que a construção de um recuperador de dados tipo STELA, pode se constituir numa empreitada que exige maior esforço técnico, que por exemplo, a construção de um DBMS (sistema gerenciador de banco de dados). Isto porque o recuperador obtêm o dado bruto e precisa tratá-lo de forma eficiente, dentro das várias alternativas de manipulação, que são as mais variadas possíveis.

A modularização que é tão aceita ao nível de aplicação, também é de necessidade vital na elaboração da programação de base. Neste caso, a modularidade foi obtida através do uso de paginação por software.

Para se obter um aconselhável índice de desempenho do produto final, foi necessário o uso de técnicas que permitem eficiente utilização de memória, baixo uso de CPU e tempo de resposta conveniente.

REFERÊNCIAS BIBLIOGRÁFICAS

1 - DAVIS, B. - The selection of Database Software - NCC Publications.

2 - Projeto BNDE/STELA - Relatório Bimestral de 15 de maio de 1976.

3 - RUDD, Walter G. - Assembly Language Programming and the IBM 360 and 370 Computers - Prentice - Hall.

4 - MARTIN, James - Computer Data-Base Organization

5 - GOOS, G., HARTMANIS, J. e BAUER, F.L. - Software Engineering - An Advanced Course

6 - OS TOTAL Reference Manual - Application Programming - Cincon Systems, Inc.

7 - C.J. Date - "An Introduction to Database Systems"

8 - KNUTH, D.E. - The Art of Computer Programming 1 - Fundamental Algoritms - Addison - Wesley, Reading, Mass, 1968 - 182 - 190

9- AUSLANDER, M.A., e JAFFE, F.F. - Influences of dynamic address translation on operating system technology - IBM Systems Journal 12 nº 4, 368 - 380, 1973.

10- FERREIRA, Marcos Alberto Neme - O Executivo do STELA-B - As características de um Monitor de Teleprocessamento.

11- Tese de Mestrado na PUC de William Carlyle Koelsch.

12- SOUZA, Flávio P. e KOELSCH, William C. - Introdução à Programação Assembler para os Sistemas 1360-1370 - (Livro ainda não editado).

13- STELA B - Manual do Usuário - Consulpuc Serviços Técnicos Ltda, Rio de Janeiro, 1977.

Apêndice A – Tabela I


```
314+*-------------------------------------------------------------------*
315+*                    TABELA I IMPRESSAO                             *
316+*-------------------------------------------------------------------*
317+ITAB     DSECT
318+ORDI     DS    C .                    ORDEM DE OCORRENCIA
319+NVARI    DS    C .                    NUMERO DO ARQUIVO
320+SNIVI    DS    C .                    SUBNIVEL DE OCORRENCIA
321+OSENTI   DS    C .                    ORDEM DE ENTRADA
322+NIVI     DS    C .                    NIVEL DE IMPRESSAO
323+FLAGSI   DS    C .                    INDICADORES
324+*
325+CALCFLGI EQU   X'80'
326+NOPRINTF EQU   X'40'
327+*DIVCFLAG EQU   X'20'
328+STRFLAG  EQU   X'10'
329+TEMTOTIF EQU   X'08'
330+TOTIFLAG EQU   X'04'
331+SECIDFLG EQU   X'02'
332+DONEFLAG EQU   X'01'
333+*
334+LOCVALFI DS    CL2 .                  LOCALIZACAO DO VALOR
335+LOCVALDI DS    C .                    LOCALIZACAO NA LINHA IMPPESSAO
336+FORMATDI DS    C .                    FORMATO PEDIDO PARA IMPRESSAO
337+*
338+FNOSPEC  EQU   X'00' .                FORMATO NAO ESPECIFICADO
339+FCOMPAC  EQU   X'01' .                FORMATO HORIZ. COMPACTADO
340+FHORIZ   EQU   X'02' .                FORMATO HORIZONTAL
341+FCENTR   EQU   X'04' .                FORMATO CENTRALIZADO
342+FVERT    EQU   X'08' .                FORMATO VERTICAL
343+*
344+LINBITS  EQU   X'70'
345+LINBIT01 EQU   X'10'
346+*
347+TAMMAXI  DS    C .                    TAMANHO DE IMPRESSAO VERTICAL
348+TAMTITI  DS    C .                    TAMANHO DO NOME USUARIO
349+NOMEI    DS    CL20 .                 NOME USUARIO DO CAMPO
350+NOMEACCI DS    CL8 .                  NOME DE ACESSO AO CAMPO
351+TAMVALFI DS    CL2 .                  TAMANHO DO CAMPO ARMAZENADO
352+TAMVALDI DS    CL2 .                  TAMANHO DE IMPRESSAO
353+TPFI    DS    C .                    TAMANHO DA PARTE FRACIONARIA
354+TIPOVALI DS    C .                    TIPO DO VALOR ARMAZENADO
355+TIPCAMPI DS    C .                  TIPO DO CAMPO
356+TIPOCMPI EQU   TIPCAMPI .            P/ SELECAO
357+*
358+TIPOCTRL EQU   X'01' .                TIPO CHAVE DE ACESSO
359+TIPOVAR  EQU   X'02' .                CAMPO DO VARIAVEL
360+TIPOLINK EQU   X'04' .                CAMPO DE LIGACAO
361+TIPOTRAD EQU   X'08' .                TIPO DA TRADUCAO
362+TRADFLAG EQU   X'10' .                CAMPO COM TRADUCAO
363+*
364+TAMPREFI DS    C .                    TAMANHO DO PREFIXO P/ TRADUCAO
365+VLINKI   DS    H .                    LINK PARA VALORES
366+       ORG   VLINKI
367+LOCTITI  DS    C .                    LOCAL PARA O TITULO
368+       DS    C
369+LOCPREFI DS    H .                    DESL. DO PREFIXO NA WORKAREA
370+OFFTABI  EQU   0
371+TAMTABI  EQU   4096
372+DELTAI   EQU   *-ITAB
373+       ORG   VLINKI
374+PREFIXO  DS    CL8 .                  PREFIXO PARA TRADUCAO
375+DELTAINC EQU   *-ITAB-DELTAI
376+TABETAPI CSECT
```

Apêndice B – Tabela R

PAGE

STMT SOURCE STATEMENT 26 NOV

```
378+*-------------------------------------------------------------------*
379+*                    TABELA R ARQUIVOS                              *
380+*-------------------------------------------------------------------*
381+RTAB     DSECT
382+NARQR    DS    CL4 .          NOME DO ARQUIVO
383+LKPATHR  DS    CL8 .          LINKPATH DO MESTRE AO VARIAVEL
384+LOCELR   DS    CL2 .          LOCALIZACAO DA ELEMENT-LIST
385+LOCIOAR  DS    CL2 .          LOCALIZACAO DA IOAREA
386+TAMIOAR  DS    CL2 .          TAMANHO DA IOAREA
387+DELTAR   EQU   *-RTAB
388+OFFTABR  EQU   0
389+TAMTABR  EQU   16*DELTAR+16
390+TABETAP1 CSECT
```

Apêndice C – Dicionário de campos

NOME USUARIO	NOME DE ACESSO	TIPO DE CAMPO	TIPO DE VALOR	TAMANHO ARMAZENADO	TAMANHO IMPRESSAO	TPF	TRADUCAO PREFIXO
BENE	BENECCNT	3	C00	011	011	00	
BENECAUS	BENECAUS	2	C00	002	002	00	
BENECCNT	BENECCNT	3	C00	011	011	00	
BENECAEX	BENECAEX	2	C40	006	006	00	
BENECAIN	BENECAIN	2	C40	008	008	00	
BENECANA	BENECANA	2	C00	008	008	00	
BENEESCI	BENEESCI	2	C00	001	001	00	
BENEINVA	BENEINVA	2	C00	001	001	00	
BENENCME	BENENCME	2	C00	048	048	00	
BENENCRD	BENENCRD	2	128	003	003	00	
BENEPARE	BENEPARE	2	C00	002	002	00	
BENESEXO	BENESEXO	2	C00	001	001	00	
BENEZATU	BENEZATU	2	128	008	008	00	
CAUSA-EXTINCAO	BENECAUS	2	C00	002	002	00	
CPF-CCNT-NO-BENEFIC	BENECCNT	3	C00	011	011	00	
DATA-EXTINCAO	BENECAEX	2	C40	006	006	00	
DATA-INSCFICAO	BENECAIN	2	C40	008	008	00	
DATA-NASCIMENTO	BENECANA	2	C00	008	008	00	
ESTADO-CIVIL	BENEESCI	2	C00	001	001	00	
INVALIDO-FISICAM	BENEINVA	2	C00	001	001	00	
NOME-BENEFICIARIO	BENENCME	2	C00	048	048	00	
NUM-ORDEM-BENEFIC	BENENCRD	2	128	003	003	00	
PARENTESCO	BENEPARE	2	C00	002	002	00	
SEXO	BENESEXO	2	C00	001	001	00	
ULTIMA-ATUALIZACA.01	BENEZATU	2	128	007	008	00	

Apêndice D - Tabela de Pivôs

- Tamanho do pivô - 2 bytes
- Localização do pivô - 1 byte
- Nome do pivô - 20 bytes
- Tamanho destino do pivô - 2 bytes
- Tamanho da parte ficcionária do pivô - 1 byte
- Tipo do pivô - 1 byte
- Flag do pivô - 1 byte

Apêndice E - Exemplos de Tabulação

```
ENTRE PEDIDO DE ACESSO :
PEDIDO DE ACESSO DO USUARIO : STELA -- ACEITO
                DATA  5 ABR 78    HORA 19:24:38

SELECIONE PROJ #
EFETUANDO PESQUISA SEQUENCIAL
SELECAO TERMINADA
FICHAS LIDAS...:        47          FICHAS SELECIONADAS...:
SUBFICHAS LIDAS:         0          SUBFICHAS SELECIONADAS:

TOTALIZE VALOR.GLOBAL POR ESTADO E POR SETOR #
IMP POR ESTADO POR SETOR TOT.VALOR.GLOBAL #
```

ESTADO	SETOR	..TOT.VALOR.GLOBAL...
AM	1300	135.000
	1400	10.000
	3000	33.000
	4100	188.000
	5000	216.000
	5100	188.000
	6000	75.000
	6403	135.000
	7100	110.000
BA	3000	60.000
	4100	145.000
	6117	145.000
DF	3000	86.000
GO	5200	20.000
	6101	216.000
	6202	145.000
MA	7200	216.000
MG	3300	63.000
	6103	216.000
PA	6117	108.000
PE	6101	108.000
	6407	20.000
RJ	1100	161.000
	1400	145.000
	3100	161.000
	5100	91.000
	6000	161.000
	6101	135.000
	6102	21.000
	6109	41.000
	6111	135.000
	6501	110.000
	7000	161.000
	7100	245.000
RS	6105	33.000
	6401	0
SP	2300	20.000
	3000	75.000
	3500	245.000
	6101	46.000
	6102	75.000
	6200	33.000

```
ENTRE PEDIDO DE ACESSO
PEDIDO DE ACESSO DO USUARIO : STELA - ACEITO
                  DATA 22 MAR 78    HORA 11:14:13

CONTE PROJ POR ESTADO E POR SETOR #
IMP POR ESTADO POR SETOR NUM.PROJ #
```

ESTADO	SETOR	NUM.PROJ
AM	1300	1
	1400	2
	3000	1
	4100	1
	5000	1
	5100	1
	6000	2
	6403	1
	7100	2
BA	3000	1
	4100	1
	6117	1
DF	3000	1
GO	5200	1
	6101	1
	6202	1
MA	7200	1
MG	3300	1
	6103	1
PA	6117	1
PE	6101	1
	6407	1
RJ	1100	1
	1400	1
	3100	1
	5100	1
	6000	1
	6101	1
	6102	2
	6109	1
	6111	1
	6501	1
	7000	1
	7100	1
RS	6105	1
	6401	1
SP	2300	1
	3000	1
	3500	1
	6101	1
	6102	1
	6200	1
	6401	1

Apêndice F- LISTAGEM DOS PROGRAMAS

```
         PRINT NOGEN
[illegible] START     [illegible],WKALOC,LCT,DEBUG=SIM
WKALOC   DSECT
NPIV     DS    H
NTOT     DS    H
CONTFLAG EQU   X'20'
TOTAFLAG EQU   X'08'
PIVFLAG  EQU   X'04'
VERBFLAG DS    X
**
*    [illegible] ANALISA O TEXTO DA MENSAGEM E CARREGA A TABEL
*    [illegible] COM OS PIVOS DA TABULACAO E OS CAMPOS CORRES
*    [illegible]
**
*    A ESTA ALTURA O PRE-PROCESSADOR JA IDENTIFICOU NO TEXTO
*    [illegible] (CONTE OU TOTALIZE)  E DESVIOU PARA TABJI
TABETAP1 CSECT
*
II       [illegible]
TABJLO   [illegible] AREAS=(A1,A2,A3)         PAGINAS DE 4K USADAS N
         [illegible] DEBUG=NAO
         USING ITAB,II
         USING WKALOC,[illegible]
         L     BASE?,[illegible]
         LA    II,NEWIELEM
           USING     WKALOC,BASED
           L         [illegible]
           LA        II,NEWIELEM
         TESTPALV PELOC,PETAM,CONTE,EQ=TAB1    VERIFICA SE VE
         B     TABIERR[illegible]
         CLI   ERRBYTE,GOOD               VERIFICA SE PROCNOME FUNC
         TESTPALV  PELOC,PETAM,TOTALIZE,EQ=TAB2
         B         TABIERR1
TAB1     MVI   VERBFLAG,CONTFLAG
BUSCA    PROXPALV  PTR,PELOC,PETAM,FIMCOM
PROCUR   PROCNOME   II,AREAPROC,NEWIELEM,PELOC,PETAM
***
*     FUNCOES DAS MACROS USADAS
*     TESTPALV  -  COMPARA PALAVRA DO TEXTO COM A PALAVRA
*                  INDICADA COMO PARAMETRO DA MACRO
*     PROXPALV  -  RECUPERA PROXIMA PALAVRA NO TEXTO
*                  CRITERIO DE SEPARACAO ENTRE PALAVRAS - BR
*     PROCNOME  -  PESQUISA CAMPO DO TEXTO NO DICIONARIO DE
*                  E COPIA ELEMENTO CORRESPONDENTE NA TABELA
***
         CLI       ERRBYTE,GOOD
         BE    ARMFLAG1
         BE    TABILOOP
         CLI   ERRBYTE,NAOENC
         BE    NAONODI1
```

```
         B     ERRONPR1
TABILOOP PROXPALV    PTR,PELOC,PETAM,FIMCOM
           CLI        0(PELOC),C'='
         BE    TABIEXIT
         TESTPALV    PELOC,PETAM,POP,NE=TABIERR2
DEPURA   DEBUG
         PROXPALV    PTR,PELOC,PETAM,FIMCOM
PROCUR1  PROCNOME    II,AREAPROC,NEWIELEM,PELOC,PETAM
         CLI   ERRBYTE,GOOD
         BE    ARMFLAG2
         CLI   ERRBYTE,NAOENC
         BE    NAONODI2
         B     ERRONPR2
ARMFLAG1 MVC   FLAGSI,VERBFLAG
         B     TABILOOP
ARMFLAG2 MVI   FLAGSI,PIVFLAG
         B     TABILOOP
NAONODI1 CALCERRO   ERRO=201,RETURN=PROCUR,RETERR=TABIERRX,MO
ERRONPR1 CALCERRO   ERRO=202,RETURN=PROCUR,RETERR=TABIERRX,M
NAONODI2 CALCERRO   ERRO=201,RETURN=PROCUR1,RETERR=TABIERRX,
ERRONPR2 CALCERRO   ERRO=202,RETURN=PROCUR1,RETERR=TABIERRX,M
TABIERR1 DEBUG
         SEND  ERRO=501
         B     TABIERRX
TABIERR2 DEBUG
         SEND  ERRO=502
         B     TABIERRX
TABIERRX DEBUG
         NI    PRFLAGS,255-TABF             DESLIGA FLAG DE TAB
         PE
TAB2     MVI   VERBFLAG,TOTAFLAG
         B           BUSCA
TABIEXIT TC   TAB020,,V
         ROTERR
         INTERPE
         ENDMOD
         END
         PRINT NOGEN
TABETAP2 STARTMOD    RTAB,ITAB,CTAB,CLLIST,WKALOC,DEBUG=SIM
WKALOC     DSECT
NPIV       DS    H
NTOT    DS    H
CONTFLAG EQU   X'20'
TOTAFLAG EQU   X'08'
PIVFLAG   EQU  X'04'
VERBFLAG DS    X
         DS    0F
ESQMLSTX DS    4X
ESQINITX DS    6X
TABETAP2 CSECT
```

```
TAB030   DENTRY    AREAS=(A1,A2,A3,D0)
         USING ELLIST,WK0
           USING       ITAB,WK1
         USING  CTAB,WK2
           USING       RTAB,WK3
           USING       WKALOC,BASE3
           RESTTAB     I,BASET1,0
           COPYWORK    BASEDADO,BASET3,WK0
           L           BASE3,DISCDADO
           LR          WK0,BASE3
           AH          WK0,WORKDESL
           ST          WK0,WORKLOC
  ELEMLIST WORKLOC,WK0,WK1,WK2,WK3,NEWTELEM,SAVEAE,REGS=(WK4
         ST      WK0,WORKLOC
         SR      WK0,BASE3
         STH     WK0,WORKDESL
           DROP        WK0,WK1,WK2,WK3
           RESLVLOC    R,WORKLOC,0,1,,REGS=(WK0,WK1,WK2,WK3,WK
         USING RTAB,WK3
         RESTRTAB    R,BASET2,0,WK3
         MVC ESQMESTX(2),=H'2'   .  2 BYTES DE PARAMETROS
         MVC     ESQMESIX+2(2),LOCCLR   .  DESL DO ESQUEMA DO M
         MVC     ESQINITX(2),=H'4'    .  4 BYTES  DE PARAMETRO
         MVC     ESQINITX+2(2),LOCCLR   . DESL DO PRIMEIRO ESQU
         XC      ESQINITX+4(2),ESQINITX+4    .  NUMERO DE VARIA
           LH          WK0,WORKDESL
           LA          WK0,31(WK0)
           SRA         WK0,5
           SLA         WK0,5
           STH         WK0,IDESTAB1
           XC          IDESL,IDESL
           LA          WK0,IDESIZE(WK0)
           STH         WK0,WORKDESL
           TC          TAB030,,V
IDESIZE    EQU         512
           LTORG
           ENDMON
         END
         PRINT  NOGEN
TABITAP3 STARTMOD  RTAB,ITAB,WKALOC
WKALOC     DSECT
NPIV       DS      H
NTOT    DS     H
CONTFLAG EQU     X'20'
TOTAFLAG EQU     X'10'
PIVFLAG   EQU    X'02'
VERSFLAG DS      X
         DS      0F
ESQMESIX DS      4X
ESQINITX DS      6X
```

```
PTABDESL DS     H
PIVTAB   DSECT
TAMPIVT  DS     X
         DS     X
LOCPIVT  DS     H
NOMPIVT  DS     CL20
TAMPPIVT DS     CL2
TPEPIVT  DS     C
TIPOPIVT DS     C
FLAGPIVT DS     C
TAMTPIVT DS     C
DELTAPIV EQU    *-PIVTAB
TABITAP3 CSECT
TAB030   DENTRY  AREAS=(A1,D0)
         USING WKALOC,BASE3
II         EQU        WK1
         USING      ITAB,II
           USING      RTAB,WK3
           XC         NPIV,NPIV
           XC         NTOT,NTOT
*        INICIALIZACAO DA AREA
         LH     WK7,WORKDESL
         LA     WK7,3(WK7)     . ARREDONDAR PARA MULTIPLO DE 4
         SRL    WK7,2
         SLL    WK7,2
         STH    WK7,PTABDESL
         AR     WK7,BASE3         .  ENDERECO REAL DA AREA DISPO
         USING PIVTAB,WK7
         RESTRTAB    I,BASET1,0
VOLTA1   NXTINTAB    I,II,BASET1,0,DELTAI
         CLI         ORDI,ORDPOSM
         BNL         CONT1
         TM          FLAGSI,PIVFLAG
         BZ          VOLTA1
         LH     WK6,NPIV
         AH     WK6,=H'1'
         STH    WK6,NPIV
         BAL    RRET,CAPPIV
           B          VOLTA1
         EJECT
CONT1    RESTRTAB    I,BASET1,0
VOLTA2   NXTINTAB    I,II,BASET1,0,DELTAI
         CLI         ORDI,ORDPOSM
         BNL         CONT2
         TM          FLAGSI,CONTFLAG
         BZ          VOLTA2
         LH     WK6,NTOT
         AH     WK6,=H'1'
         STH    WK6,NTOT
         BAL    RRET,CAPPIV
```

```
         B              VOLTAL
*  NESTE PONTO  O VALOR DE  FLAGSI  E X'03'
CONTZ      [illegible]TAB    I,BASET1,0
VOLTAB     WKFINTAB    I,II,BASET1,0,DELTAT
         CLI         TPRI,ONEPOSM
         BNE         [illegible]
         TM          FLAGSI,TOTAFLAG
         BZ          VOLTAL
         LH     WK6,N[illegible]
         AH     WK6,=H'1'
         STH    WK6,N[illegible]
         BAL    [illegible],CALPIV
         B           VOLTAL
CONTD    SR     WK7,BASE3
         STH    WK7,[illegible]ROESL
         BR
         [illegible]
CALPIV   MVC    TAMPIVT,TAMVALFIFE
         MVC    LOCPIVT,LOCVALFI
         MVC    [illegible]PIVT,[illegible]
         MVC    TI[illegible]IVT,TI[illegible]VALI
         MVC    TPRPIVT,TPRI
         MVC    FL[illegible]IVT,FLAGSI
         MVC    T[illegible]PIVT,TAMVALDI
         MVC    TAM[illegible]IVT,[illegible]
         LA     WK7,[illegible]PIV(WK7)
         BR     [illegible]
         [illegible]
         [illegible]
         END
         [illegible]
TABLTAP4 START [illegible]   ITAP,WKALOC,ELLIST,DEBUG=SIM
WKALOC   DSECT
IPIV     DS     H
[illegible]     DS     H
CONTFLAG EQU    X'[illegible]'
TOTAFLAG EQU    X'[illegible]'
PIVFLAG  EQU    X'0A'
[illegible]LAC DS     [illegible]
         DS     [illegible]
ESQ[illegible]X DS     [illegible]X
ESQ[illegible]X DS     [illegible]Y
PTABDESL DS     H
INIC     DS     H
[illegible]C    DS     H
PROX[illegible]P DS     H
[illegible]    DS     PL3
TABLTAP4 CSECT
[illegible]    [illegible]Y AREAS=(A3,LU)
         USING  WKALOC,BASE3
BASEWORK EQU    BASET3
         LA     RPRM,ESQINITX
         TCR    INIFEAD,((RPRM)),R,S
         MVC    INIC,MARCAF
         XC     PROXDISP,PROXDISP
LER      DS     0H
         LA     RPRM,ESQMESTX
         TCR    READMES,((RPRM)),R,S
         B      FIMMEST
         USING  PIVTAB,IPIV
         USING  TVALEXT,I
         LH     NV,NPIV
         LA     I,INIC
         LR     IPIV,BASE3
         AH     IPIV,PTABDESL
VOLTA    DS     0H
*                         NA PRIMEIRA VEZ    LIRMAOX=INIC
         TM     0(I),X'40'
         BO     ACFESCEN
         LR     WK0,BASE3                .   LOC    DA IDAREA
         AH     WK0,LOCPIVT
         XR     WK1,WK1
         IC     WK1,TAMPIVT
         L      PROXR,BASEWORK
         AH     PROXR,0(I)
         BCTR   WK1,0
         B      *+10
         CLC    VALTABX-TVALEXT(*-*,PROXR),0(WK0)
         EX     WK1,*-6
         BH     ACFESCEN
         BL     MENOR
         LA     IPIV,DELTAPT(IPIV)
         SH     NV,=H'1'
         LA     I,LFILHOX-TVALEXT(PROXR)
         BZ     SOMA
         B      VOLTA
MENOR    LA     I,LIRMAOX-TVALEXT(PROXR)
         B      VOLTA
SOMA     TM     FLAGPIVT,TOTAFLAG
         BO     SOMATOT
         L      PROXR,BASEWORK
         AH     PROXR,0(I)
         ICM    WK0,B'1111',0(PROXR)
         AH     WK0,=H'1'
         STCM   WK0,B'1111',0(PROXR)
         B      LER
SOMATOT  LA     RPRM,TABTEMP
         SR     RPRM,BASE3
         STH    RPRM,SOMATP1
         OI     SOMATP1,BASE3*16
```

```
         MVC   SOMATP2,LOCPIVT
   NAO E' PROCESSAR IO  MOVER  TPF  AQUI
         MVC   SOMATP3,TIPOPIVT
         MVC   SOMATP4,TAMPIVT
         L     R15,MVDECP
         BALR  RRET,R15
SOMATP1  DC    H'0'
SOMATP2  DC    H'0'
SOMATP3  DC    X'0'
SOMATP4  DC    X'0'
         MVC   SOMATP21,TPFPIVT
         MVC   SOMATP23,TPFPIVT
         MVC   SOMATP22,SOMATP1
         MVC   SOMATP24,0(I)
         L     R15,=V(BASEWORK)
         OI    SOMATP24,RPNM*16
         L     R15,=V(PROTUS)
         BALR  RRET,R15
SOMATP21 DC    X'0'
SOMATP22 DC    X'0'
SOMATP23 DC    X'0'
SOMATP24 DC    X'0'
         B     LER
INSCONT  TM    FLAGPIVT,TUTAFLAG
         BO    INSTOT
         L     PROXR,BASEWORK
           LR  WK0,PROXR
           AH  WK0,=H'4096'
           AH  PROXR,PROXDISP
           LA  WK1,4(PROXR)
           XR  WK2,WK2
           IC  WK2,TAMPIVT
           BNL OVERFLOW
           MVC 0(4,PROXR),=F'1'
           S   PROXR,BASEWORK
           STH PROXR,0(I)
           LA  WK0,4(PROXR)
           STH WK0,PROXDISP
           B   LER
INSTOT   L     PROXR,BASEWORK
         LR    WK0,PROXR
         AH    WK0,=H'4096'
         AH    PROXR,PROXDISP
         LA    WK1,4(PROXR)
         CR    WK1,WK0
         BNL   OVERFLOW
         MVC   INSTOTP1,PROXDISP
         L     R15,=V(BASEWORK)
         OI    INSTOTP1,RPNM*16
         MVC   INSTOTP2,LOCPIVT
*  NAO E' PROCESSAR IO  MOVER  TPF  AQUI
         MVC   INSTOTP3,TIPOPIVT
         MVC   INSTOTP4,TAMPIVT
         L     R15,MVDECP                MOVE PARA 8 BYTES DEC
         BALR  RRET,R15                  EM  PROXDISP + RPPM
INSTOTP1 DC    H'0'                      BASE + DESLOC DESTINO
INSTOTP2 DC    H'0'                      TPF + DESLOC FONTE
INSTOTP3 DC    X'0'                      TIPO VAL FONTE
INSTOTP4 DC    X'0'                      TAM  VAL FONTE
         LH    PROXR,PROXDISP
         STH   PROXR,0(I)
         LA    PROXR,8(PROXR)
         STH   PROXR,PROXDISP
         B     LER
         SPACE 3
LFILDIF  DC    AL2(LFILHOX-TVALEXT)
         SPACE 3
ACRESCEN L     PROXR,BASEWORK
*                        TESTAR  SE JA' SUFICIENTE NA AR
           LR  WK0,PROXR
           AH  WK0,=H'4096'
         AH    PROXR,PROXDISP
         XR    WK2,WK2
         IC    WK2,TAMPIVT
*                              FALTOU  O TAMANHO DO VALOR
         LA    WK1,DELTATX(PROXR,WK2)
           CR  WK1,WK0
           BNL OVERFLOW
*                              TESTA  ESTOURO DA AREA
*                              AGORA  INICIALIZA NOVO ELEMENT
           MVC LIRMAOX-TVALEXT(L'LIRMAOX,PROXR),0(I)
         TM    LIRMAOX-TVALEXT(PROXR),X'FF'
         BNO   NAOPAI
         LR    WK0,I
         S     WK0,BASEWORK
         SH    WK0,LFILDIF  COMPENSA DIF ENTRE LFILHOX E COM
         STH   WK0,LIRMAOX-TVALEXT(PROXR)
         OI    LIRMAOX-TVALEXT(PROXR),X'80'
NAOPAI   MVC   LFILHOX-TVALEXT(L'LFILHOX,PROXR),MARCAF
           MVI CAITTABX-TVALEXT(PROXR),CRITICAL
         MVC TAMTABX-TVALEXT(L'TAMTABX,PROXR),TAMPIVT
         LR    WK0,BASE3          INC  DA IOAREA
           AH  WK0,LOCPIVT
           XR  WK1,WK1
           IC  WK1,TAMPIVT
           BCTR  WK1,0
           B   *+10
           MVC  VALTABX-TVALEXT(*-*,PROXR),0(WK0)
           EX   WK1,*-6
*  EXECUTAR  MVC ACIMA SUBSTITUINDO O TAMANHO DE  WK1  NO SE
*    (TAMANHO - 1)
```

```
         MVC   0(2,1),PROXDISP
       LA    WKO,DELTATXEI(WK1,PROXR)
*                        DEVEMOS  ARREDONDAR  PROXR
         S       WKO,BASEWORK
         STH     WKO,PROXDISP
PROXNIVA  LA     IPIV,DELTAPT(IPIV)
         SH      IV,=H'1'
        LA     I,LFILHOX-TVALEXT(PROXP)
         B       INSCONT
*
         B       VOLTA
OVERFLOW MAJORMSG  *C1
         RC
FIMREST  TM   ,  TABCFO,,V
         SPACE 3
PIVTAB   DSECT
TAMPIVT      DS    X
         DS    X
LOCPIVT      DS    H
NOMPIVT  DS      CL20
TAMOPIVT DS      CL
IPRPIVT  DS      C
ITERPIVT DS      C
FLAGPIVT DS      C
TAMTPIVT DS      C
DELTAPT  EQU     *-PIVTAB
         SPACE 3
TVALEXT      DSECT
             DS      0F
LINMAOX      DS      H
LFILHOX      DS      H
CRITERIOX    DS      C
TAMIADX  DS      X
DELTATX      EQU     *-TVALEXT
VALIADX  DS      C
TABETAP4 CSECT
*  DEFINICAO DE CONSTANTES
*  REGISTRADORES
MARCAF   DC      X'FFFF'
WZ       EQU     WK0
I        EQU     WK5
PROXR    EQU     WK4
IPIV     EQU     WK2
CRITICAL     EQU     X'C1'
         ENTRY
         END
         PRINT NOGEN
TABETAP5 STARTMOD ITAB,WKALOC
WKALOC   DSECT
NPIV     DS      H
```

```
NTOT     DS      H
CONTFLAG EQU     X'20'
TOTAFLAG EQU     X'02'
PIVFLAG  EQU     X'04'
VERBFLAG DS      X
         DS      0F
ESUMASTX DS      4X
ESSINITX DS      6X
PTABDESL DS      H
INIC     DS      H
ELMLOC   DS      H
PROXDISP DS      H
TABTEMP  DS      PL8
GDCOD    DS      X
SAVI     DS      H
SAVIV    DS      H
TABETAP5 CSECT
         SPACE 3
TAB050   DENTRY AREAS=(43,DO)
         PRINT GEN
BASEWORK EQU     BASET3
         USING WKALOC,BASE3
         USING PIVTAB,IPIV
         LA      WKO,2048+16           CONSERTAR ERRO QUE DESCON
         STH     WKO,WORKDESL          EXISTE NA ETAPA 4
         MVI     GDCOD,X'00'
         LH      NINDI,NPIV
         AH      NINDI,NTOT
         L       IPIV,BASEDADO
         AH      IPIV,PTABDESL
         LH      WKO,WORKDESL
         LA      WKO,DELTPTP )(WKO)
         STH     WKO,PTRVAZO
         USING CABTOL,WKO
         USING PIVTAB,IPIV
CICLO1   TM      FLAGPIVT,PIVFLAG
         BZ      CONCONT
         IC      WKO,GDCOD
         LA      WKO,X'10'(WKO)
         STC     WKO,GDCOD
         LH      WKO,PTRVAZO
         A       WKO,BASEDADO
         MVC     CODCAB,GDCOD
         XR      WK1,WK1
         IC      WK1,TAMTPIVT
         BCT     WK1,*+10
         MVC     NOMCAB(*-*),NOMPIVT
         EX      WK1,*-6
         LA      WK1,1(WK1)
         XR      WK2,WK2
```

```
         IC    WK2,TAMPIVT
         STC   WK2,TAMVAL
         LA    WK2,DELTACAB(WK1)
         STC   WK2,TAMCAB
         AR    WK0,WK2
         MVC   T[illegible]CAB(4),TAMDPIVT
         B     C[illegible]
CODCONT  LH    WK0,PTRVAZO
         A     WK0,BASEDADO
         MVC   CODCAB,GDCOD
         XR    WK2,WK2
         IC    WK2,TAMTPIVT
         LA    WK2,=H'[illegible]'
         SRA   [illegible]
         LA    WK2,[illegible]
         BCT   WK2,[illegible]
         MVC   N[illegible]+4(*-*),NUMPIVT
         EX    WK2,*-6
         LA    WK2,1+4(WK1)
         TM    FLAGPIVT,TOTAFLAG
         BZ    NUMCAB
         MVC   NOMCAB(4),=C'TOT.'
         LA    WK2,8
         B     CABCOMUM
NUMCAB   MVC   NOMCAB(4),=L'NUM.'
         LA    WK2,4
CABCOMUM STC   WK2,TAMVAL
         LA    WK2,DELTACAB(WK1)
         STC   WK2,TAMCAB
         AR    WK0,WK2
         TM    FLAGPIVT,TOTAFLAG
         BO    TOTCAB
         MVC   T[illegible]CAB(4),NTIPONUM
         B     C[illegible]
TOTCAB   MVC   TAMCAB,TAMDPIVT
         MVC   TIPOCAB,TIPOPIVT
         NI    TIPOCAB,X'EC'
         OI    TIPOCAB,X'A0'
         MVC   TPFCAB,TPFPIVT
*        B     C[illegible]
CODCOMUM SR    WK0,WK1
         LH    WK1,PTRVAZO
         XR    WK2,WK2
         IC    WK2,TAMCAB
         AR    WK1,WK2
         IC    WK2,TAMVAL
         AR    WK1,WK2
         STH   WK1,PTRVAZO
         LA    IPIV,DELTAPIV(IPIV)
         BCT   NINDI,CICLO1

         MVI   GDCOD,X'F0'
CODMBAS  LH    WK0,PTRVAZO
         A     WK0,BASEDADO
         MVC   CODCAB,GDCOD
         MVC   NOMCAB(4),MESTBASE
         MVI   TAMVAL,X'FF'
         LA    WK2,DELTACAB+4
         STC   WK2,TAMCAB
         MVC   TAMDCAB+4(4),=X'00FF0000'
         LA    WK2,X'FF'(WK2)
         AH    WK2,PTRVAZO
         MVI   GDCOD,X'00'
         LR    WK0,WK2
         A     WK0,BASEDADO
         MVC   CODCAB,GDCOD
         LA    WK2,1+2(WK2)             TAMANHO TOTAL =
         SH    WK2,=H'12'                  PTRVAZO + 1 + 2 - 12 -
         SH    WK2,WORKDESL
         L     WK0,BASEDADO
         AH    WK0,WORKDESL
         STH   WK2,10(WK0)              ARMAZENA TAM ANTE CABECAL
         LA    WK0,4095
         STH   WK0,PTRVAZO
         OI    OPDFLAG,PRIMFILA+NEWFILA
         NI    OPDFLAG,X'FF'-(FIMFILA+FIMSDFT)
SEGUE    LH    I,INIC
         USING TVALEXT,I
         USING TDL,WK0
         L     IPIV,BASEDADO
         AH    IPIV,PTABDESL
         USING PIVTAB,IPIV
         LH    NV,NPIV
         MVI   GDCOD,X'10'
TESTA    DS    0H
DESV1    DS    0H
VALPIV   XR    WK1,WK1
         IC    WK1,TAMPIVT
         AH    WK1,PTRVAZO
         SH    WK1,WORKDESL
         AH    WK1,=H'1'
         LA    WK2,512
         CR    WK1,WK2
         BL    SEGUE1
         S     IPIV,BASEDADO
         STH   NV,SAVNV
         BAL   RRET,GRAVABUF
         LH    NV,SAVNV
         A     IPIV,BASEDADO
         LA    WK0,DELTPTRO
         AH    WK0,WORKDESL
```

```
         STH   WK0,PTRVAZO
SEGUE1   LH    WK0,PTRVAZO
         A     WK0,BASEDADO
         A     I,BASET3
         MVC   CODIOL,GDCOD
         XR    WK1,WK1
         IC    WK1,TAMPIVT
         BCT   WK1,*+10
         MVC   VALIOL(*-*),VALTABX
         EX    WK1,*-6
         AH    WK1,=H'1'
         SH    WK1,PTRVAZO
         STH   WK1,PTRVAZO
         SH    NV,=H'1'
         BZ    SEGUE2
         IC    WK0,GDCOD
         LA    WK0,X'10'(WK0)
         STC   WK0,GDCOD
         LA    IPIV,DELTAPIV(IPIV)
         LH    I,LFILHOX
         B     DESV)
SEGUE2   S     I,BASET3
         STH   I,SAVI
         A     I,BASET3
         LH    I,LFILHOX
         A     I,BASET3
         LA    IPIV,DELTAPIV(IPIV)
         LH    NT,NTOT
VALCONT  TM    FLAGPIVT,TOTAFLAG
         BZ    ROT4
         LH    WK1,PTRVAZO
         AH    WK1,=H'8'
         B     SEGUE3
ROT4     LH    WK1,PTRVAZO
         AH    WK1,=H'5'
SEGUE3   LA    WK0,012
         SH    WK1,WORKDESL
         LA    WK1,WK2
         BL    SEGUE4
         S     IPIV,BASEDADO
         S     I,BASET3
         STH   NV,SAVNV
         BAL   RRET,GRAVABUF
         LH    NV,SAVNV
         A     I,BASET3
         A     IPIV,BASEDADO
         LA    WK0,DELTPTRO
         SH    WK0,WORKDESL
         STH   WK0,PTRVAZO

SEGUE4   LH    WK0,PTRVAZO
         A     WK0,BASEDADO
         MVC   CODIOL,GDCOD
         TM    FLAGPIVT,TOTAFLAG
         BZ    ROTI4
         LA    WK1,8
         B     SEGUE5
ROTI4    LA    WK1,4
SEGUE5   BCT   WK1,*+10
         MVC   VALIOL(*-*),VALCONTX
         EX    WK1,*-6
         LA    WK1,1+1(WK1)          COMPENSAR BCTR + 1 PARA C
         LH    WK2,PTRVAZO
         AR    WK2,WK1
         STH   WK2,PTRVAZO
         AR    I,WK1
SEGUE7   SH    NT,=H'1'
         BZ    SEGUE8
         LA    IPIV,DELTAPIV(IPIV)
         B     VALCONT
SEGUE8   DS    0H
         LH    I,SAVI
         A     I,BASET3
TESTFLAG LA    NV,1(NV)
         TM    LIRMADX,X'40'
         BZ    ROTIRMAO
         CH    NV,NPIV
         BNL   TEST1
         LH    I,LIRMADX
         SH    I,=H'16384'
         A     I,BASET3
         XR    WK0,WK0
         IC    WK0,GDCOD
         SH    WK0,=H'16'
         STC   WK0,GDCOD
         B     TESTFLAG
ROTIRMAO LH    I,LIRMADX
         LA    WK1,DELTAPIV
         LH    WK0,NPIV
         SR    WK0,NV
         MR    WK0,WK0               WK0,WK1=WK0 * WK1
         AH    WK1,PTABDESL
         A     WK1,BASEDADO
         LR    IPIV,WK1   IPIV=BASEDADO+PTABDESL+DELTAPIV(NPI
         B     TESTA
TEST1    OI    ORDFLAG,FIMFILA+FIMSORT
         LH    WK0,PTRVAZO
         A     WK0,BASEDADO
         MVI   CODIOL,X'00'          ASSINALA FIM IDLIST
         BAL   RRET,GRAVABUF
```

```
         DC    IDLCOM,IDLCOM
         AZ    NOFREE
         R.EFID  IDLCOM,IDLFIM
NOFREE   MVC   IDLCOM,NEWTTRF
         MVC   IDLFIM,TTR
         MVC   MRSIDBLK,MESTBASE
         BC
         EJECT
*
*        GRAVABUF DESTPOI WK0,WK6 E PODE MUDAR PAGINA
*
GRAVABUF DS    0H
         USING BUFFER,BF
         SR    RRET,BASE1
         STH   RRET,SVINS
         L     WK0,ORDFLAG
NOFLAG   BLSUG (WK0),P00
         TM    ORDFLAG,PRIMFILA
         BZ    GRAVAB02
         MVC   NEWTTRF,NULO
         L     WK0,=Y(TTR-WKALOC)
         BLEKN BASEDADO,(WK0),ERR=ESTAB01
         MVC   TTRANT,TTR
GRAVAB02 TM    ORDFLAG,FIMSORT
         BO    GRAVAB04
         L     WK0,=Y(TTR-WKALOC)
         BLEKN BASEDADO,(WK0),ERR=ESTAB01
GRAVAB04 L     BF,BASEDADO
         LH    BF,WORKDESL
         TM    ORDFLAG,NEWFILA
         BZ    GRAVAB05
         MVC   TTRPROXF,NEWTTRF
         MVC   NEWTTRF,TTRANT
         B     GRAVAB06
GRAVAB05 MVC   TTRPROXF,=X'FFFFFFFF'
GRAVAB06 LH    WK0,PTRVAZO
         A     WK0,BASEDADO
         TM    ORDFLAG,FIMFILA
         BZ    GRAVAB07
         MVC   TTRPROXB,NULO
         MVI   0(WK0),X'00'
         B     GRAVAB08
GRAVAB07 MVC   TTRPROXB,TTR
         MVI   0(WK0),X'01'
GRAVAB08 BLEKREG  BASEDADO,WORKDESL,TTRANT
         L     WK0,BASEDADO
         LH    WK0,WORKDESL
GRAVBUF= BLSUG (WK0),512,R
         NI    ORDFLAG,255-NEWFILA-FIMFILA-PRIMFILA
         MVC   TTRANT,TTR
         LH    RRET,SVINS
         AR    RRET,BASE1
         BR    RRET
ESTAB01  SEND  ERRO=600
         RC
         DS    OF
* CONSTANTES E REGISTRADORES
NULO     DC    X'FF000000'
MAICAR   DC    X'FFFF'
TIPOBFIX EQU   X'C0'
TIPODCOM EQU   X'A8'
         DS    OF
KTIPUNUM DC    XL3'000800'
         DC    AL1(TIPOBFIX)
KTIPUTOT DC    XL3'001500'
         DC    AL1(TIPODCOM)
NV       EQU   WK6
I        EQU   WK5
IPIV     EQU   WK4
NINDI    EQU   WK7
NT       EQU   WK7
         LTORG
         DSECT
PIVTAB   DSECT
TAMPIVT  DS    X
         DS    X
LOCPIVT  DS    H
NOMPIVT  DS    CL20
TAMDPIVT DS    CL2
TPFPIVT  DS    C
TIPOPIVT DS    C
FLAGPIVT DS    C
TAMTPIVT DS    C
DELTAPIV EQU   *-PIVTAB
         SPACE 3
BUFFER   DSECT
TTRPROXB DS    F
TTRPROXF DS    F
NUMREG1  DS    H
NUMREG2  DS    H
DELTPTFO EQU   *-BUFFER
ELEMBUF  DS    0H
         SPACE 3
CABIDL   DSECT
         DS    OF
CODCAB   DS    X
TAMCAB   DS    X
TAMVAL   DS    C
NOMCAB   DS    0CL20
TAMDCAB  DS    CL2
```

```
[illegible]  DS      C
[illegible]  DS      C
DELTACAB  EQU     *-CABIDL
          SPACE   3
IDL       DSECT
          DS      OF
[illegible]  DS   X
DELTAIDL  EQU     *-IDL
VALIDL    DS      OX
          SPACE   3
TVALEXT   DSECT
[illegible]  DS   OF
[illegible]  DS   H
[illegible]  DS   H
[illegible]  DS   C
[illegible]  DS   X
DELTATX   EQU     *-TVALEXT
VALT[illegible]  DS  C
TABETAPS  DSECT
          [illegible]
          END
          GETWKM BASEDADO,(WK0),ERR=ESTAB01
```

```
          [illegible]
          AR      RRET,BASE1
          BR      RRET
ESTAB01   SEND    ERRO=600
          RC
          DS      OF
* CONSTANTES E REGISTRADORES
NULO      DC      X'FF000000'
MARCAF    DC      X'FFFF'
TIPOBFIX  EQU     X'C0'
TIPODCOM  EQU     X'A8'
          DS      OF
KTIPONUM  DC      XL3'000800'
          DC      AL1(TIPOBFIX)
KTIPOTOT  DC      XL3'001500'
          DC      AL1(TIPODCOM)
NV        EQU     WK6
I         EQU     WK5
IPIV      EQU     WK4
NINDI     EQU     WK7
NT        EQU     WK7
          LTORG
```